# SuperX-Adminstrationshandbuch Kernmodul

- Daniel Quathamer
  danielq@memtext.de
- Meikel Bisping
  mbisping@memtext.de


http://www.superx-projekt.de


Version 4.0rc1
Stand 1.9.2011


Lehrfilm zur Installation von Postgres

Lehrfilm zur Installation des Kernmoduls

**Lizenz Java**

# Inhaltsverzeichnis

# 1Einführung

Das Berichtssystem SuperX ist ein sog. *Data Warehouse* für Bildungseinrichtungen, d.h. beliebig viele Datenquellen werden unter einer einheitlichen Auswertungsschnittstelle zur Verfügung gestellt. Da jede Hochschule unterschiedliche Datenquellen besitzt und in SuperX übernehmen will, bereiten wir für jede Datenquelle ein Modul vor, z.B. ein COB-Modul oder ein SOS-Modul. Bei Bedarf können Anwender auch eigene Module für proprietäre Datenquellen erzeugen und SuperX so erweitern.

Die Module enthalten die wichtigsten Prozeduren, Tabellen und Abfragen für die jeweilige Datenquelle. Der Startpunkt ist das Kernmodul. Eine Kurzanleitung für die Installation ist vorbereitet.

| Zur Geschichte von SuperX |
| --- |
| SuperX wurde in den 90er Jahren an der Universität Karlsruhe von der Projektgruppe Abakus unter der Leitung von Herbert W. Roebke entwickelt. *SuperX* stand damals für: **S**ystem zur **U**nterstützung von **P**lanung und **E**ntscheidung des **R**ektorats durch **I**nformation, **C**ontrolling und **S**imulation. In der damaligen Version, die im Folgenden als das "alte SuperX" bezeichnet wird, bestand das System aus einer SuperX-Datenbank (Informix) und einem Win32 / SGI / Mac-Client. Im Zuge der Verbreitung von WWW-basierten Frontends wurde im Jahr 2000 in Karlsruhe der Client nach Java portiert. Dieses SuperX-Applet wurde an der Universität Duisburg weiterentwickelt und aus Performance- und Sicherheitsgründen in eine Applet-Servlet-Anwendung (3-tier) geändert. Da SuperX sich vor allem dann als nutzbar erwiesen hat, wenn das Berichtssystem auf die Bedürfnisse der Anwender (in der Regel Hochschulen) zugeschnitten werden kann, ist die neue SuperX-Anwendung ein Open Source-Projekt, d.h. Anwender können die Datenbank und den Client für ihre Zwecke ändern. Lizenzrechtlich basiert SuperX auf der CampusSource-Lizenz , einer Variante der GPL (http://www.campussource.de/lizenz).<br>Um die Installation und die Weiterentwicklung von SuperX überschaubar zu halten, hat die Projektgruppe SuperX in Duisburg Ende 2001 beschlossen, die Datenbank in Module zu zerlegen. Das vorliegende Kernmodul soll sicherstellen, dass das gesamte System selbst nach Änderung von anderen Modulen weiterhin lauffähig und übertragbar bleibt.<br>Die vorliegende Dokumentation wird außerdem deutlich machen, dass die "neue" Architektur einige Änderungen an der SuperX-Datenbank erfordert. Bisherige Anwender der Karlsruher SuperX-Anwendung erhalten eine spezielle Anleitung für das Update. |

Falls es bei der Implementation des Kernmoduls zu Problemen kommt, können Sie sich unter www.superx-projekt.de informieren. Oder mailen Sie uns direkt: danielq@memtext.de bzw. mbisping@memtext.de.

## 1.1Sicherheitsaspekte

Da SuperX für den Einsatz in großen Netzen konzipiert wurde, sind folgende Schutzmechanismen implementiert:

- Benutzer- und Paßwortkontrolle
- SHA-Verschlüsselung von Passwörtern
- Zugriffsprotokollierung
- Benutzerspezifische Einschränkung des Angebots an Abfragemasken
- Benutzerspezifische Einschränkung der einsehbaren Institutionen in Datenbankanfragen
- Getrennte Datenhaltung (operative Systeme - SuperX)

- Abschottung der Datenbank gegenüber fremden Zugriffen (z.B. mit ODBC) durch 3-Tier-Architektur. Auch Client-Anwendungen wie das Informix Client SDK werden nicht eingesetzt.
- Verschlüsselte Verbindung von Client und Servlet (https), Möglichkeit der Zwischenschaltung eines Apache-Webservers (ggf. in der DMZ).
- Bei Einsatz des Applets: Zusätzliche Verschlüsselung der in der Anwendung eingebaute Applet/Servlet-Kommunikation

Nur das SuperX-Servlet auf dem Webserver und die SuperX-Datenbankadministratoren auf der Serverseite haben einen direkten Zugriff auf die SuperX-Datenbank. Alle anderen Zugriffsmöglichkeiten für Benutzer können ausgeschlossen werden, d.h. kein Zugriff mit anderen SQL-Frontend-Programmen wie ISQL, DBACCESS (Unix) oder ODBC (Windows, Mac).

### 1.1.1Notiz zum SuperX-Applet

Das SuperX-Applet war das Standard-Clientprogramm von SuperX in den Versionen 2.1 bis 3.0final. Theoretisch besteht bei diesem "Rich Client" die Gefahr des Missbrauchs durch Modifikation der (frei verfügbaren) Quellen. Daher ist das Applet wg. Sicherheitsproblemen nur noch im geschützten Intranet nutzbar, bzw. für Entwicklungszwecke.

Hinzu kommt, dass das Java-Applet die Installation einer (relativ schwergewichtigen) Java Runtime erforderlich macht, was im betrieblichen Einsatz häufig zu Problemen führt. Nach dem Boom von Java in den  Jahren 1999 bis 2004 ist mit dem Stichwort "Ajax" eine neue Technologie aufgekommen, die sowohl für die Bedienung als auch für die Administration zeitgemäßer ist.

Die Funktionalität des Applets wird durch den Ajax-Client im XML-Frontend ersetzt. Das Applet lässt sich abschalten .

## 1.2Erforderliche Hardware

Im Minimalbetrieb ist das gesamte SuperX-System auf einem Desktop-PC  installierbar, z.B. auf einem SuSE-Linux 8.1-PC; dies reicht für den Testbetrieb im Intranet mit wenigen Usern vollkommen aus.

Für den Einsatz im Echtbetrieb unterscheiden wir mindestens drei Komponenten:

- Ein Datenbankserver
- Ein Java-basierter Web- und Applicationserver
- Die SuperX-Clients

Für jede Komponente gibt es unterschiedliche Empfehlungen.

### 1.2.1Datenbankserver

Wir empfehlen generell Intel-Architektur mit Linux als Betriebssystem, da dies relativ kostengünstig ist und immer weitere Verbreitung findet. SuperX benötigt mindestens Java 1.6 (eine neuere Version ist empfehlenswert) und bash-2.x-Scripte auf dem DB-Server einsetzen, beides läuft sicher unter Linux.

**DB-Server**

- Intel-Architektur
- 2GHZ-Prozessor, z.B. AMD Athlon oder Intel Xeon
  (Dual-Processor-System für Informix-Betrieb besonders sinnvoll)
- 512 kb L2-Cache
- 1 GB DDR-Ram
- RAID-Controller
- Festplatte IDE oder SCSI, 7200 U/Min, UDMA/133
  1-6 GB Kapazität für DB

Wir empfehlen die Hochleistungsserver aus den aktuellen Produktpaletten von HP, Dell, Sun oder vergleichbaren Herstellern, die Firma Novell/SuSE zertifiziert auch Hardware für Linux. Für den produktiven Einsatz empfehlen wir einen Mittelklasse-Server mit dem Betriebssystem Linux. SuperX benötigt an einer größeren Hochschule (>10.000 Studierende, viele HIS-Systeme) erfahrungsgemäß 1-2 GB Platz für den DB-Server.

### 1.2.2Webserver

Wirt empfehlen auch hier Intel-Architektur und Linux als Betriebssystem für Neuanschaffungen. Wenn bei SuperX nur das Applet als Frontend verwendet wird, sind die Anforderungen gering: Ein Server mit 1- 2 GHZ-Prozessor, 512 MB RAM reichen völlig aus.

Der Webserver benötigt keinen Plattenplatz, aber eine leistungsfähige CPU. Die RAM-Anforderungen auf dem Webserver sind geringer, da wir mit einem Connection Pool arbeiten. Der SuperX-Webserver läßt sich auch ideal auf bereits konfigurierten Webservern (z.B. Homepages von Hochschulverwaltungen) einsetzen, sofern nicht bereits Tomcat oder ein anderer Application Server darauf läuft.

Wenn das XML-Frontend eingesetzt werden soll, empfiehlt sich eine etwas leistungsfähigere Architektur, ggf. sogar der Betrieb von zwei Webservern im software-basierten Lastausgleich ("load balancing" via Tomcat).

**Web-Server**

- Intel-Architektur
- 2GHZ-Prozessor, z.B. AMD Opteron, Athlon oder Intel Xeon
- 512 kb L2-Cache
- 2 GB DDR-Ram
- RAID-Controller
- Festplatte IDE oder SCSI, 7200 U/Min, UDMA/133
  Kaum Festplatten-Kapazität notwendig

Generell gilt natürlich die Devise: So viel CPU-Taktfrequenz und RAM wie Sie sich leisten können.

Je nach Architektur der Umgebung sollte der Webserver 2 Netzwerkkarten enthalten, eine davon ist mit dem DB-Server verbunden, die andere mit dem Firewall-Rechner. Alternativ kann der Webserver auch in einer DMZ stehen, dann reicht eine Netzwerkkarte.

## 1.3Erforderliche Software

Die SuperX-Datenbank läuft auf Windows- und Linux-Rechnern. Der SuperX-Client läuft auf allen Plattformen, die die Java-Runtimeumgebung (1.6.x) anbieten. Für den DB-Server empfehlen wir in jedem Falle einen UNIX bzw LINUX-Server, da alle serverseitigen Scripte als Shellscripte konfiguriert sind. SuSE Linux (Server-System mit Linux-Kernel, Version 2.2.16 oder höher, z.B. SuSE Linux 8) versteht sich gut mit Informix, aber alle aktuellen Linux-Distributionen enthalten bereits Java, Tomcat und Postgres.

Beim Informix-Datenbankserver sollten Sie darauf achten, daß SuperX in einem eigenen Online-System läuft.

Die erforderliche Software für den Betrieb des Kernmoduls:

| **Software** | **Win32-Systeme** | **Linux / AIX** | **MacOS X** |
|---|---|---|---|
| **DB-Server** | | | |
| SuperX-Datenbank | PostgreSQL 7.2.X oder höher (in Verbindung mit Cygwin) | Informix Dynamic Server f. Unix 7.3 oder höher bzw. PostgreSQL 7.2.X oder höher | PostgreSQL 7.2.X oder höher - |
| **Webserver** | | | |
| Webserver | • Apache oder<br>• IIS 4.0 / 5.0 oder<br>• (beliebig) | • Apache oder<br>• (beliebig) | (beliebig) |
| Servlet-Engine | • Tomcat 5.x oder höher | • Tomcat 5.x oder höher | (beliebig, sollte aber Verschlüsselung bieten) |
| Java | mindestens SUN/Oracle JDK 1.6.x | mindestens SUN/Oracle JDK 1.6.x | JDK 1.6.x-i (nur für MacOS X, nicht mehr für MacOS 8.x oder 9.x verfügbar) |
| **Client** | | | |
| Webbrowser | IE6, Firefox2 und höher oder andere, sollten aber Verschlüsselung bieten | Firefox2 und höher oder andere, sollten aber Verschlüsselung bieten | Firefox2 und höher oder andere, sollten aber Verschlüsselung bieten |
| Java-Runtime (nur bei Einsatz des Applets) | mindestens JRE 1.6.x | mindestens JRE 1.6.x | JRE 1.3.x-i (nur für MacOS X, nicht mehr für MacOS 8.x oder 9.x verfügbar) |

Noch ein Hinweis zur Zeichen-Codierung: Mit dem Kernmodul 4.0 ist neben der ISO-Codierung auch UTF-8 möglich. Achten Sie darauf, das jeweils passende SuperX-Paket herunterzuladen (im Dateinamen befindet sich entweder "iso" oder "utf8"). Weitere Hinweise siehe Kapitel zur Zeichencodierung.

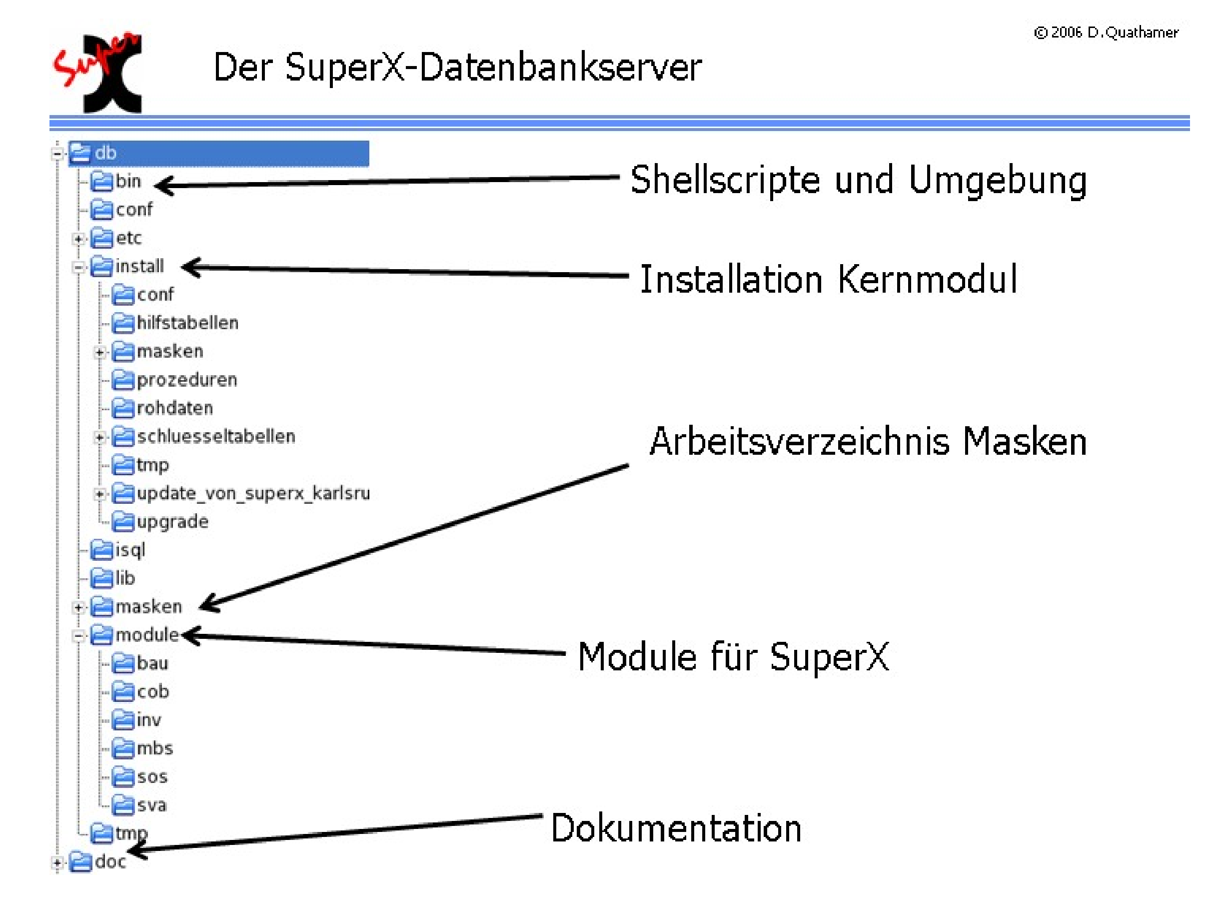
## 1.4Das Kernmodul

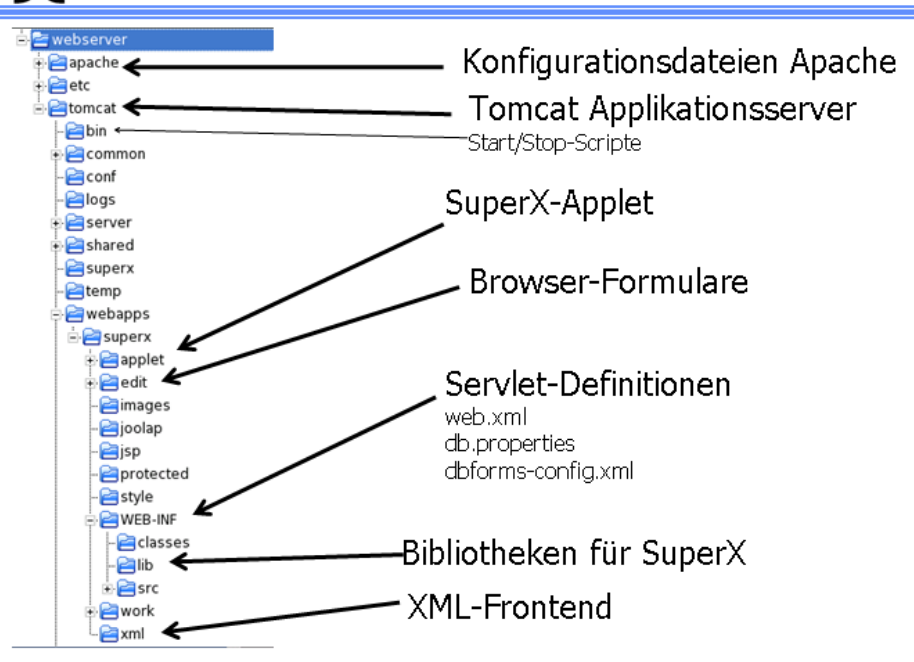
Das SuperX-Kernmodul beinhaltet alle zum Betrieb von SuperX unbedingt notwendigen Tabellen, Prozeduren und Abfragen; die wichtigsten Tabellen werden unten näher beschrieben.

Die folgende Tabelle zeigt die Ordnerstruktur des Kernmoduls auf einen Blick:

| Pfad | | Beschreibung |
|---|---|---|
| db | | Die SuperX-Datenbankseite |
| | bin | Shellscripte |
| | etc | Beispiel-Initscripte für SuperX-DB-Dienste |
| | install | Installationsscripte |
| | isql | isql-Formulare, Scripte und Berichte |
| | masken | Entladene Masken |
| | module | Modulpfad |
| doc | | Dokumentation |
| src | | Java-Quellen de.superx.* |
| webserver | tomcat | Tomcat-Beispielimplementation (Tomcat 3.2.2) |
| | apache | Apache-mod_jk (binär für SuSE Linux 8-9+ source) |
| | etc | Beispiel-Initscripte für SuperX-DB-Dienste |

Die folgenden Abbildungen zeigen die Ordnerstruktur von jeweils Datenbank-Seite und Webserver-Seite.






Der Datenbankserver kann auf einem anderen Rechner liegen als der Webserver; es ist aber auch möglich, das gesamte SuperX auf einem Rechner zu installieren. Je nach Hardware- oder Softwarevoraussetzungen kann dies ein WinNT/2000- oder Linux-Rechner sein. Unter Windows können Sie z.B. ein Verzeichnis `C:\superx` erstellen; unter Linux sollten Sie einen Nutzer `superx` mit dem Verzeichnis `/home/superx` einrichten. Den von Ihnen gewählten Pfad bezeichnen wir als im Folgenden als `$SUPERX_DIR`, und alle Verzeichnisse des Kernmoduls (`db`,`doc`,`webserver`) werden dort hineinkopiert.

Die Rohdaten der Module liegen in einem eigenen Unterverzeichnis `rohdaten` unterhalb des Modul-Verzeichnisses[1]. Ggf. ist es zweckmäßig, aus dem Modulpfad einen symbolischen Link auf den Entladepfad vorzunehmen, z.B. im Pfad `db/module/sva` geben Sie ein:

```
ln --symbolic <<Tatsächlicher Entladepfad>> rohdaten
```

## 1.5Ausbaustufen einer SuperX-Implementierung

SuperX liefert eine datenbankbasierten Website zur Präsentation von Inhalten der Hochschule für die öffentliche Nutzung im Internet sowie für die interne Nutzung im Intranet. Nach einer Datenübernahme aus den operativen Systemen gilt es, eine effiziente Berichterstellung zu ermöglichen und Export- und Importschnittstellen zu bieten. Das System wird in mehreren Aufbaustufen realisiert, wichtig ist daher die Skalierbarkeit des Systems vom Prototypen bis zum Echtbetrieb.

Das zu realisierende System besteht aus drei Komponenten: der Datenbank, der Webanwendung und des Clients (3-tier-Application). Die folgende Abbildung zeigt eine typsiche Beispielarchitektur:

[1] in älteren Versionen des SuperX-Kernmoduls laden die Rohdaten unter `db/rohdaten`. Dies hat sich als unpraktisch erwiesen

Die Clients im Intranet greifen direkt oder über die Webanwendung auf die Datenbank zu. Die Clients im Internet greifen über den Browser (http oder für Verschlüsselte Zugänge https) auf die Inhalte zu.

Durch diese Architektur wird verhindert, dass WWW-Clients direkten Zugriff zur Datenbank haben. Bei mittlerer Last ist diese Architektur ausreichend.

Falls die Last ansteigt, ist das System wie folgt skalierbar:

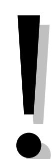
Die SuperX-Datenbank wird angebunden an ein oder mehrere operative Vorsysteme. Gleichzeitig, um die Webanwendung zu entlasten, ist es möglich sein, die Last auf einen zweiten Webserver auszulagern ("Load balancing").

# 2Installation

Die Installationsschritte beziehen sich auf die Neuinstallation und das Upgrade. Für die Neuinstallation gibt es eine Kurzanleitung unter Linux.

## 2.1Neuinstallation

Bei der Neuinstallation können Sie einfach alle Komponenten in einen Pfad `$SUPERX_DIR` kopieren und von dort die unten genannten Installationsschritte durchführen. Beim Update können Sie die Patchdatei in `$SUPERX_DIR` entpacken; die "alten" Dateien werden ersetzt. Wenn Sie die Datenbank und den WWW-Server auf getrennten Systemen betreiben, dann entpacken Sie am besten die Update-Datei in einem temporären Verzeichnis und kopieren dann die Ordner `/db` und `/webserver` auf die entsprechenden Rechner.



Wichtig:Ändern Sie bitte keinesfalls die Ordnerstruktur unterhalb von `/db` und `/webserver`; Sie können u.U. keine Updates ohne umfangreiche Anpassungen einspielen. Besonders bei der Inbetriebnahme des Systems ist es für die Fehlersuche unerläßlich, die Ordnerstruktur einzuhalten.

SuperX ist zwar ein sehr offenes System, aber gewisse Konventionen werden sich in Zukunft als nützlich erweisen, wenn verschiedene Hochschulen Daten und Scripte austauschen wollen. In jedem Fall

empfehlen wir Ihnen immer erst dann manuelle Anpassungen, wenn die Anwendung oder das Script funktioniert – eine äußerst sinnvolle Heuristik für die Arbeit mit derart komplexen Systemen wie SuperX.

### 2.1.1Übersicht über Installationsschritte

Das Kernmodul wird in drei Arbeitsschritten installiert:

- Installation und Einrichtung der Datenbank
- Installation eines Webservers mit Servlet-Engine
- Installation der Java Runtime auf den Clients (nur bei Einsatz des Applets)

Die folgende Übersicht zeigt das Vorgehen bei der SuperX-Installation, darauf folgt eine Kurzanleitung für die Installationsmaßnahmen:

| Schritt | Erläuterung |
|---|---|
| Kopieren und Vorbereiten des Kernmoduls nach $SUPERX_DIR | Bringen Sie das SuperX-Kernmodul in ein Verzeichnis auf dem Rechner, am besten auf den Datenbankserver; ggf. können Sie die Verzeichnisse `/doc` und `/webserver` auf einen anderen Rechner verschieben.<br>Unter Windows können Sie z.B. ein Verzeichnis `C:\superx` erstellen und unter Linux einen Nutzer `superx` mit dem Verzeichnis `/home/superx` einrichten und alle Verzeichnisse des Kernmoduls (`db,doc,webserver`) dort hineinkopieren. Bei Betrieb unter Windows muss das gesamte db-Verzeichnis auf einen UNIX-Rechner verschoben werden.<br>Kopeiren Sie die Datei `$SUPERX_DIR/db/bin/SQL_ENV.sam` nach `$SUPERX_DIR/db/bin/SQL_ENV` und passen Sie die Umgebungsvariablen an. |
| Installation der erforderlichen Software | • Installieren Sie auf dem Datenbankserver Informix oder PostgreSQL<br>• Installieren Sie auf dem Webserver Java die Datenbanktreiber, und setzen Sie die Umgebungsvariable JAVA_HOME für Tomcat in der Datei `$SUPERX_DIR/db/bin/SQL_ENV`<br>• Installieren Sie auf den Clients die Java-Runtime und die Javahilfe |
| Start des Datenbankservers | Starten Sie den Datenbankserver und spielen Sie danach die SuperX-Datenbank des Kernmoduls ein; die Zugangsparameter müssen Sie dem SuperX-Servlet bekannt geben. |
| Start des SuperX-Servlets | Gehen Sie in das Verzeichnis /webserver/tomcat/bin und starten Sie Tomcat, ggf. als Dienst |
| Test des Webservers | Öffnen Sie die Datei `http://<<Rechnername>>:8080/superx/` im Browser und testen Sie zuerst die Anmeldung im XML-Frontend (und dann ggf. im Applet, wenn Sie dies nutzen wollen) |
| Freigabe des Webservers im Netz | Nun ist die Basisinstallation vom Kernmodul abgeschlossen, und Sie können mit der Anpassung für Ihre Einrichtung beginnen. Richten Sie User ein, und geben Sie die WWW-Adresse Ihres Webservers im Intra- oder Internet (in diesem Fall benötigen Sie auch die die Verschlüsselung) frei. |
| Einspielen der Module | Füllen Sie SuperX mit den einzelnen Modulen; bisherige SuperX-Andender können ihr bisheriges System übernehmen. |

### 2.1.2Besonderheiten für verschiedene Betriebssysteme

Wir empfehlen den Einsatz von SuperX unter Linux. Für andere Betriebssysteme gelten hier und da Besonderheiten.

#### 2.1.2.1Windows / Cygwin

Unter Windows lassen sich derzeit der Applikaitonsserver von SuperX und Postgres betreiben. Der Datenbankserver läßt sich nur betreiben, wenn vorher die Unix-Emulation Cygwin installiert wird und wenn Postgres als Datenbanksystem gewählt wird.

Die aktuelle Cygwin-Distribution erhalten Sie von www.cygwin.com, das genaue Vorgehen haben wir bei der Installation von PostgreSQL beschrieben. Erfahrungsgemäß ist die Postgres-Version in Cygwin aktueller als in einer "normalen" Linux-Distribution.

#### 2.1.2.2AIX / HP-UX

SuperX läuft unter den Systemen AIX und HP-UX, mit folgender Vorbedingung: Die Shellscripte in SuperX benötigen eine `bash`-Shell sowie eine installiertes Java JDK 1.6.x oder höher. Bei AIX ist keine SUN-Java-Runtime verfügbar, so dass eine Verschlüsselungsklasse nachinstalliert werden muss. Kopieren Sie die Datei `sunjce-provider.jar` von einer Windows- oder Linux-Installation in den System-Klassenpfad der IBM-JAVA-Installation (z.B. nach `lib/ext`).

#### 2.1.2.3Noch nicht getestete Betriebssysteme

Folgende Betriebssysteme wurden bisher noch nicht als Plattfomen für SuperX getestet:

- Solaris
- MacOS X

### 2.1.3 Kurzanleitung: Das Vorgehen -kurz und knapp für Linux-Systeme

Lehrfilm zur Installation des Kernmoduls

Achtung: der Film zeigt noch die Installation unter ISO-Codierung, dies ist mit dme Kernmodul 4.0 nicht mehr nötig.

**Voraussetzungen**

Postgres bzw. Informix IDS ist gestartet und läuft, der User existiert im Datenbanksystem hat das Recht, Datenbanken zu erzeugen, Java ist installiert.Auf dem Server sollte kein weiterer Server-Dienst auf den Ports 8005, 8009 und 8080 (Tomcat-Standard-Ports) laufen.

**Entpacken**

Entpacken Sie das Kernmodul in /home/superx

```
tar -xzvf
kern<<Versionsnr>>_superx_<<Codierung>>_<<DBMS>>.tar.gz
```

**Umgebungsvariablen**

Gehen Sie in das Verzeichnis db/bin

```
cd db/bin
```

Kopieren Sie SQL_ENV.sam nach SQL_ENV

```
cp SQL_ENV.sam SQL_ENV
```

Wenn Sie die Datenbank superx unter Linux in /home/superx mit Postgres als DB-Server und Java im Verzeichnis /usr/lib/java installiert haben, brauchen Sie nichts ändern.

Ansonsten passen Sie $SUPERX_DIR, $JAVA_HOME, $DATABASE, $SX_CLIENT, $LANG und $MAILPROG an

Starten Sie das Script mit

```
. SQL_ENV
```

und fügen Sie den Aufruf am Ende der Datei ~/.bashrc ein:

```
. ~/db/bin/SQL_ENV
```

**Einspielen der Datenbank**

Gehen Sie in das Install-Verzeichnis
`cd $SUPERX_DIR/db/install`
Starten Sie das Script
`kernmodul_erzeugen.x <<ggf. mit Name des DBSpace>>`
Bei Fehlern kommt eine Meldung, Protokolle stehen in
`create.log`
Damit ist die db-Seite eingerichtet.

**Steuerungsdatei für das Servlet: db.properties**

Nun gehen Sie in das Verzeichnis
`cd $SUPERX_DIR/webserver/tomcat/webapps/superx/WEB-INF`
Kopieren Sie `db-postgres.properties` bzw. `db-informix.properties` nach `db.properties`
Beispiel für Postgres:
`cp db-postgres.properties db.properties`
Staten Sie den PropAdmin mit
`propadmin.x`
Passen Sie hier den Servernamen, Datenbanknamen, Usernamen und Passwort an, und drücken Sie "Verbindung testen"
Danach speichern Sie die Datei.

**Connection Pool für dbforms**

Kopieren Sie die Datei Sie die Datei `SUPERX_DIR/webserver/tomcat/webapps/superx/META-INF/context.xml.sam` nach context.xml und editieren Sie die Datei. Passen Sie im Abschnitt `<ResourceParams name="jdbc/superx">` die Verbindungsparameter an, die Sie oben auch dem SuperX-Servlet gegeben haben (also `driverClassName` und `url`).

| | |
|---|---|
| **Steuerungsdatei für das Applet: superx.properties** | Wenn Sie das Applet benutzen wollen:<br>Nun gehen Sie in das Verzeichnis<br>`$SUPERX_DIR/webserver/tomcat/webapps/superx/applet`<br>`cd $SUPERX_DIR/webserver/tomcat/webapps/superx/applet`<br>Kopieren Sie die Datei `superx-postgres.properties.sam` bzw. `superx-informix.properties` nach<br>`superx.properties`<br>`cp superx-postgres.properties superx.properties`<br>Editieren Sie die Datei, und tragen Sie bei `SxTitle` den Hochschulnamen ein.<br>Wenn Sie Informix benutzen, muss bei SxDB "`Informix`" stehen, bei Postgres "Postgres" |
| **Start von Tomcat** | Nun wechseln Sie in das Verzeichnis<br>`cd $SUPERX_DIR/webserver/tomcat/bin`<br>Und starten tomcat mit<br>`startup.sh`<br>(Stop übrigens mit "`shutdown.sh`".) |
| **Und los geht's...** | Nach erfolgreichem Start rufen Sie im Browser auf<br>`http://localhost:8080/superx/`<br>Dort können Sie das Applet und das XML-Frontend aufrufen. Geben Sie bei der Kennung `superx` ein, und als Passwort "anfang12" |
| **Feineinstellung** | Melden Sie sich im XML-Frontend an und gehen Sie im Menü auf "Tabelle suchen", klicken einfach "Abschicken" und öffnen Sie die Tabelle hochschulinfo; dort sollten Sie Ihre Hochschule auswählen und "Speichern" anklicken, damit Ihre Hochschulnummer gespeichert wird. |

### 2.1.4Installation und Pflege der SuperX-Datenbank

Die SuperX-Datenbank liegt als exportierte Datei in dieser Distribution vor und kann einfach importiert werden. Zunächst muss aber der Datenbankserver eingerichtet werden. Derzeit laufen die Installationsscripte und auch alle Modulscripte nur unter UNIX /Linux / Cygwin. Bei Betrieb von SuperX unter Windows muss also das gesamte Verzeichnis `db` auf einen UNIX-Rechner kopiert und betrieben werden. Deshalb empfehlen wir für den Anfang einen Linux-Rechner, da hier die Java-Unterstützung kein Problem ist.

#### 2.1.4.1Einrichten des Datenbankservers unter UNIX / LINUX

Der Datenbankserver läuft unter Informix (mind. Version 7.31) und PostgreSQL (mind. Version 7.2).

##### 2.1.4.1.1Stopp: welche Zeichencodierung soll es werden?

Bevor Sie das Kernmodul entpacken, sollten Sie sich vergewissern dass die Zeichencodierung des jew. Pakets mit der installierten übereinstimmt.

Die auf Ihrem System installierte Codierung erfahren Sie mit dem Befehl

```
echo $LANG
```

Mögliche Ausgaben sind `de_DE@euro` (oder die jew. Variante mit ISO) und `de_DE.utf8` (je nach UNIX gibt es hier unterschiedliche Schreibweisen, z.B. auch `de_DE.UTF-8`). Wenn Sie sich nicht sicher sind, welche Codierung überhaupt installiert ist, können Sie mit

```
locale -a | grep de
```

eine Liste der deutschsprachigen Locales anzeigen.

SuperX unterstützte bis Version 3.5 nur die Locale de_DE@euro. Ab Version 4.0 ist auch UTF-8 möglich. Es ist auch ein Mischbetrieb möglich: Der Datenbankserver läuft mit ISO-Codierung, und der Tomcat mit UTF-8-Codierung. Achten Sie nur darauf, dass Sie immer das Paket mit der jeweils richtigen Codierung laden.

> Hinweis für OpenSuse 11.4 und Postgres: Die psql- und Java-Shell in OpenSuse 11.4 wertet nicht nur die Variable LANG aus, sondern auch "LC_ALL". Die Ursache dafür haben wir noch nicht gefunden. Im Zweifelsfall setzen Sie LC_ALL auf den gleichen Werte wie LANG.

Mit dem SuperX-Kernmodul werden Scripte ausgeliefert, mit denen die Codierung von Dateien flexibel geändert werden kann.

#### 2.1.4.1.2User superx - Kernmodul entpacken

Legen Sie einen User `superx` am einfachsten mit dem home-Verzeichnis `/home/superx` an.

Wenn wir im Folgenden `$SUPERX_DIR` sprechen, meinen wir `/home/superx`. Es ist natürlich auch jedes andere Verzeichnis möglich.

Es muss auf Betriebssystemebene sichergestellt werden, dass das Dateisystem Textdateien in der passenden Locale anlegt sind. Bei modernen UNIXen wird die Umgebungsvariable $LANG auf "UTF-8" gesetzt.

Setzen Sie die richtige Locale z.B. mit `LANG=de_DE.utf8; export LANG`.

Entpacken Sie die `kernmodul-XX.tar.gz` imVerzeichnis $SUPERX_DIR.

Machen Sie eine Kopie der Datei Date `$SUPERX_DIR/db/bin/SQL_ENV.sam` und nennen Sie sie einfach `SQL_ENV`. In dieser Datei werden viele allgemeine Konfigurationen der Umgebung vorgenommen. Prüfen Sie, ob die in der SQL_ENV angegebene Locale (`LANG=de_DE...`) existiert.

Geben Sie der Datei ggf. Ausführungsrechte mit `chmod +x SQL_ENV`.

#### 2.1.4.1.3Informix

SuperX unter Informix läuft derzeit unter UNIX und LINUX. Für den Datenbankserver unter Windows NT benötigen Sie in jedem Fall einen UNIX / LINUX-Rechner für die Shellscripte in den Modulen. Das Vorgehen ist im Abschnitt Konfiguration beschrieben..

***2.1.4.1.3.1Systemvoraussetzungen***

Da die meisten Hochschulen bereits Informix-Datenbanken einsetzen, sind hier keine Hinweise zur Installation nötig. Da SuperX ein beliebtes System für Linux-basierte Systeme ist, hier nur ein paar kurze Hinweise für Informix 9.x  unter Linux

Informix für Linux lässt sich ab Version 7.3 unter Linux installieren (wir haben SuSE Version 7.3-8.1 und RedHat 8/9 getestet). Gemäß Anleitung von IBM/Informix geht man so vor:

1. Als root anmelden
2. User und Gruppe `informix` anlegen (achten Sie darauf, dass die Default-Gruppe des Users "informix" nicht die Gruppe "users" ist, sondern "informix").
3. Die Umgebungsvariable z.B. auf /home/informix setzen
   `export INFORMIXDIR=/home/informix`
   setzen
4. Dann die Informix- sql-CD einlegen und mounten, bzw. das IDS-Archiv in ein beliebiges Verzeichnis entpacken
5. `./ids_install` starten (Serverpaket wählen, Seriennummer etc eingeben), und zum Abschluß  auch das Script `./RUN_AS_ROOT.server`

Nur bei Informix 9.2x, nicht bei 9.3 oder höher:

6. Dann die IDS_2000-CD einlegen und mounten
7. Dann  startet man unter SUSE Linux oder RedHat9 (bei RedHat kann man den Hinweis, dass die Installation vom user informix gemacht werden sollte ignorieren)
   `rpm -i --relocate /opt/informix=/home/informix /mnt/IDS_2000/IDS.RPM`

   Unter **RedHat 8.0** existiert ein Bug im RPM-Programm[2], deshalb kann man Informix nicht nach `/home/informix`, sondern nur nach `/opt/informix` installieren und vorher die Umgebungsvariablen setzen:
   `RPM_INSTALL_PREFIX=/opt/informix`
   `INFORMIXDIR=/opt/informix`
   Danach startet man die Installation mit
   `rpm -i ids.rpm`

Damit ist der IDS installiert.

Die Bibliothek `libpthread` muss richtig eingebunden werden. Überprüfen kann man das Einbinden der Bibliotheken über

```
ldd $INFORMIXDIR/bin/oninit
```

Es muss erscheinen (vielleicht über einen symbolischen Link):

```
...
libpthread.so.0 => /lib/i686/libpthread.so.0
...
```

**9. für RED-Hat 8 und 9**

Wenn beim `oninit` die Fehlermeldung erscheint

```
oninit: relocation error: /var/lib/libpthread.so.0: symbol __on_exit, version GLIBC_2.0 not defined in file libc.so.6 with link time reference .
```

Dann muss man unter RedHat noch eine Bibliothek ändern:

I. Als root bennenen Sie den alten Link um:
`mv /var/lib/libpthread.so.0 /var/lib/libpthread.alt.`

[2] Bug: 158974 RPM INSTALLATION USING RELOCATE OPTION FAILS ON REDHAT LINUX V8 {NBS}

II. Dann erzeugen Sie einen neuen Link:

```
ln -s /lib/i686/libpthread.so.0 /var/lib/libpthread.so.0
```

Um die menübasierten Tools von Informix (dbaccess, onmonitor) zu nutzen, muss man falls eine entsprechende Fehlermeldung erscheint . die ältere libncurses.so.4 einbinden. Man prüft, wo libncurses.so.4 auf der Platte liegt und erstellt einen symbolischen Link.

```
ln -s /usr/lib/libncurses.so.5 /usr/lib/libncurses.so.4
```

#### *2.1.4.1.3.2Konfiguration*

Die Kofiguration des IDS geschieht im onmonitor über das Menü Mode->Parameters oder direkt in der Textdatei "onconfig", für unser Beispiel onconfig.superx. Die Pfade zu $INFORMIXDIR müssen ggf. angepaßt werden, die Voreinstellung ist oft "/usr/informix". Wichtig ist außerdem der DBSpace (zu prüfen mit onstat -d).

Zum Betrieb von SuperX hier nur einige Angaben zur empfohlenen Größe: Für das Kernmodul selbst würden 100 MB ausreichen, wenn Sie aber als erstes das SOS-Modul installieren möchten, sollten Sie nicht unter 400 MB starten (Parameter ROOTSIZE in onconfig.superx s.u.).

Wir empfehlen, das Logging auszuschalten, da SuperX keine Dialog-Anwendung ist und durch die Prozeduren sehr viel Logging anfallen würde. Selbst bei ausgeschaltetem Logging entstehen noch sehr viele Eintragungen, deshalb sollten Sie als Log Archive Tape Device /dev/null angeben.

Für die Rohdaten aus den operativen Systemen gibt es ein eigenes Verzeichnis, z.B. $SUPERX_DIR/db/module/<<modulname>>/rohdaten. Aus Platzgründen, und um sich den ftp-Transfer zu ersparen, bietet es sich unter UNIX an, hier NFS-Laufwerke einzurichten.

Falls Sie noch keine onconfig Datei für SuperX haben, erstellen Sie eine Kopie von /home/informix/etc/onconfig.std und nennen Sie sie onconfig.superx.

> Wenn man den DB-Space in einem Cooked-File ablegen will, kann man z.B. als root eine leere Datei /var/informix/rootdbs erstellen (z.B. leere Datei mit vi). Beim DB-Space müssen Sie darauf achten, dass der Benutzer und die Gruppe informix Schreibrechte auf den Cooked File bzw. die Datenpartition haben. Dieser Pfad muss dann als Parameter für den DB-Space in der onconfig.superx angegeben werden.

```
ROOTNAME rootdbs
ROOTPATH /var/informix/rootdbs
```

In der onconfig-Datei für SuperX sind die Parameter DBSERVERNAME (wir empfehlen superx_host) und DBSERVERALIAS (wir empfehlen superxdb) wichtig.

Entsprechend dieser zwei Parameter ergänzen Sie die Datei sqlhosts.

| **$INFORMIX/etc/sqlhosts** | Die Datei mit den Hostnamen für Shared Memory-Zugriff (statt `miles` geben Sie den in `/etc/hosts` definierten Rechnernamen an) und für TCP-Zugriff. |
|---|---|

Beispiel:

```
#*******************************************************************************
#Servername      Protokoll       Host     Service
superx_host     onipcshm        miles    superx_shm
superxdb        onsoctcp        miles    superx_server
```

Machen Sie eine Ergänzung in /etc/services

| **/etc/services** | Der SuperX-Service mit Portnummer |
|---|---|

```
#
superx_server   1542/tcp
```

Unter Informix für Windows NT befindet sich die onconfig unter `%INFORMIXDIR%\etc\onconfig`, die sqlhosts wird in der Registry unter HKEY_LOCAL_MACHINE oder besser über das Programm `setnet32` geändert.

Wichtig ist die Eintragung eines DBSERVERALIAS, über den das Servlet die Verbindung aufbaut. Der Port des Service in /etc/services wird ebenfalls benötigt.

Diese Parameter werden in der Datei db.properties vom SuperX-Servlet benötigt.

Es muss sichergestellt werden, dass einige Umgebungsvariablen beim Start initialisiert werden. Je nach UNIX-Art geschieht das in der `.profile` oder. `.bashrc` im Home-Verzeichnis der Benutzer informix und superx. (Im Zweifelsfall ausprobieren)

Damit man die Umgebungsvariabeln nur an einer Stelle zu pflegen braucht, empfiehlt es sich, dem User Informix Leserechte auf die Datei `$SUPERX_DIR` /db/bin/SQL_ENV zu geben und diese in der `.profile` bzw. `.bashrc` der beiden User aufzurufen.

Eintrag: `. $SUPERX_DIR/db/bin/SQL_ENV`

**Wichtig für den Einsatz unter Linux / Unix: die SQL_ENV unter Informix**

```
#SX_CLIENT=pgsql;
SX_CLIENT=dbaccess
SUPERX_DIR=/home/superx; export SUPERX_DIR
INFORMIXDIR=/home/informix; export INFORMIXDIR
INFORMIXSERVER=superx_host; export INFORMIXSERVER
ONCONFIG=onconfig.superx; export ONCONFIG
echo
echo "ONCONFIG:       " $ONCONFIG
echo "INFORMIXSERVER: " $INFORMIXSERVER
CLIENT_LOCALE=de_de.8859-1; export CLIENT_LOCALE
DB_LOCALE=de_de.8859-1; export DB_LOCALE
SERVER_LOCALE=de_de.8859-1; export SERVER_LOCALE
TERMCAP=$INFORMIXDIR/etc/termcap; export TERMCAP
TERM=ansi; export TERM
#Terminal für TeraTerm Pro auf Win32-Systemen:pctcp
PATH=${PATH}:${SUPERX_DIR}/db/bin:$INFORMIXDIR/bin; export
PATH
DBDELIMITER=^ export DBDELIMITER
DBDATE=DMY4.; export DBDATE
DBMONEY=.; export DBMONEY
TERMINAL=`tty`; export TERMINAL
#Wenn auf DB-Server auch Webserver / Tomcat läuft
#Beispiel für Suse Linux 7.3-Installation:
export JAVA_HOME=/usr/lib/jdk1.6.29
```

Stellen Sie sicher, dass die Zeile `#SX_CLIENT=pgsql;` mit dem Gatterzaun auskommentiert ist und die Zeile `SX_CLIENT=dbaccess` nicht;

In dieser Datei werden auch die Pfade und Parameter für das Laden der Daten aus den operativen System festgelegt. Sie wird von den Entladescripten und von den Cronjobs benutzt.

Für Informix ist es generell günstiger, unter Unix / Linux mit einem ANSI-Terminal zu arbeiten. Beachten Sie allerdings, daß bei dieser Einstellung kein xterm verfügbar ist und Sie somit keine graphischen Java –Anwendungen, z.B. den propadmin, auf dem Datenbankserver starten können.

Die Umgebungsvariablen DBTEMP und PSORT_DBTEMP sind eigentlich nicht mehr notwendig; wenn es Probleme beim Sortieren und Auslagern auf temporäre Datenträger gibt, dann sollte man diesen Pfad ebenfalls setzen. Die `onconfig.superx` liegt unter `$INFORMIXDIR/etc` und muss unbedingt als Parameter die Zeile

```
DBSPACETEMP   dbtemp                      # Default temp dbspaces
```

enhalten, wobei der Name `dbtemp` im `onmonitor` frei gewählt werden kann.

Ist die Umgebung  korrekt eingerichtet, dann startet man den IDS mit

**Erstmaliger Start des IDS**

oninit -ivy

Weitere nützliche Kommandozeilen-Befehle für Informix

| oninit | startet den Datenbankserver und bringt ihn in online-Modus |
|---|---|
| onstat | Zeit die aktuellen Prozesse des DB-Servers an |
| oninit -s | Von Offline nach quiescent |
| onmode -m | Von quiescent nach online |
| onmonitor | Zeigt aktuellen Status sowie ein Menü zur Administration an |
| tail -200 $INFORMIXDIR/online.log | Zeigt das Ende der Logdatei an |
| ipcs -m | Anzeigen von Shared Memory für die Datenbank |
| oncheck -pt <<Datenbank>>:<<Tabellen-name>> | Zeigt die Extents einer Tabelle an |
| oncheck -ce | Zeigt den genutzten Speicherbedarf der Extents für jeden dbspace an |

Dann kann man die Datenbank als User superx einspielen (s.u.).

Für den Ablauf der UNIX-Scripte zu den Masken (sx_select_mask, sx_insert_mask etc.) und für Cron-Jobs müssen die Parameter in der Datei $SUPERX_DIR/db/bin/SQL_ENV stimmen.

Hinweis für Datenbankserver unter AIX oder anderen Linux / Unix-Derivaten: Beachten Sie, daß die Scripte nur dann lauffähig sind, wenn auf dem Datenbankserver unter /bin/bash die bash Version 2.x oder höher liegt (bzw. gelinkt ist). Die Scripte von SuperX erwarten die bash-Shell im Verzeichnis /bin; wenn dies nicht der Fall ist, sollte die Datei sh z.B. von /usr/bin nach /bin kopiert oder gelinkt werden. Unter Ubuntu Linux 6.10 beispielweise ist die Standardshell nach /bin/dash gelinkt, dies müssen Sie für SuperX ändern.

**Exkurs: SuperX als Clientrechner für Basissysteme**

Informix kann auch auf einem UNIX-Rechner installiert werden, der nur als Client auf einen anderen Server zugreift, z.B. einen Informix-Server unter Windows NT. Dazu müssen die sqlhosts-Einträge auf beiden Rechnern übereinstimmen, und der Port des Service muss in /etc/services stehen. Auf dem Informix-Server muss man ggf. in der Datei /etc/hosts.equiv (unter Windows in c:\windows\system32\drivers\etc) die IP-Nummer bzw. den DNS-Namen des Client-Rechners freischalten. Der Zugriff auf den Remote-IDS-Server geht dabei nicht über Shared Memory, sondern über tcp.
Im Verzeichnis db/bin des Kernmoduls steht die Datei
SQL_ENV_fuer_remote_entladen.sam
Diese Beispieldatei für Informix zeigt, wie man den SuperX-Rechner als Client auf eine anderen Datenbankserver nutzen kann.
Dies ist nützlich, da so die Entladescripte nicht auf dem DB-Server des operativen Systems laufen, sondern auch auf dem SuperX-Rechner.

##### 2.1.4.1.4 Installation von PostgreSQL

Lehrfilm zur Installation von Postgres

SuperX ist seit Version 2.1 mit Postgres 7.2 bis 8.2 lauffähig, die neuen Module im Kernmodul 4.0 laufen auch unter Postgres 8.3 und 8.4. Die Distribution von Postgres für Unix findet sich unter www.postgresql.org. Eine Version für Windows befindet sich im Cygwin-Paket, dass Sie von unserem www.cyg-

win.com beziehen können. Eine allgemeine Anleitung befindet sich unter http://www.postgresql.org/idocs/index.php?install-upgrading.html, Spezialitäten für Cygwin finden Sie unten.

Verschiedene Linux-Distributionen enthalten zwar bereits Postgres und müssen nicht "von Hand" installiert werden, doch spätestens beim Datenbankupdate ohne die jeweils neue Distribution rächt sich dies: Die Distribution legt Postgres in anderen Verzeichnissen ab, und das Installationsscript von Postgres passt dann nicht mehr. Dieses Problem besteht unter Red Hat Linux und SuSE Linux. Wir empfehlen daher, die in die Distribution "eingebaute" Version von Postgres zunächst zu deinstallieren.

###### *2.1.4.1.4.1Neuinstallation (am Beispiel derVersion 7.3.4)*

**Voraussetzungen** Postgres läuft unter verschiedenen UNIX-Varianten, z.B. Linux, HP-UX oder MacOS X. Wir empfehlen für den Einstieg Linux[3]. Vor der Installation unter Linux sollte die Locale-Umgebungsvariable `$LANG` auf den gewünschten Wert geändert werden (`de_DE.utf8` oder `de_DE@euro` oder eine andere deutsche Locale (meist in `/usr/lib/locale`). Die aktuelle Locale wird bei der Installation von Postgres berücksichtigt und sorgt dafür, dass Datums- und Währungsformate korrekt sind.
Bei SuSE Linux 7.x bis 11.x ist es für ein Kompilieren der Postgres-Quellen erforderlich, dass die Pakete `gcc`, `glibc`, `gettext`, `gettext-devel`, `readline`, `readline-devel`, `zlib` und `zlib-devel` installiert sind.

Erzeugen Sie zunächst den User postgres mit dem Homeverzeichnis der Postgres-Installation (z.B. unter Linux mit `useradd -g users -d /usr/local/pgsql postgres`).

In der Postgres-7.2-Version ist es wichtig, den Datenbankserver für internationale Sprachumgebungen zu konfigurieren, deshalb bietet es sich an, die Installation nicht als Binary- sondern als Quell-Installation vorzunehmen[4]. Bei höheren Versionen von Postgres ist dies nicht mehr nötig.

In der Download-Version von Postgres wird Postgres standardmäßig nach `/usr/local/pgsql` installiert. Als DBSpace muss man ein oder mehrere Verzeichnisse anlegen und mit `initdb` vorbereiten. Die SuperX-Datenbank läßt sich dann in einem eigenen DBSpace ablegen.

Zunächst müssen Sie sich als root anmelden. Wir gehen im folgenden davon aus, dass die Quellen von Postgres im Verzeichnis

```
/usr/src/packages/SOURCES
```

liegen (das Archiv z.B. von `postgresql-7.3.4.tar.gz` muss hier entpackt werden).

Dann gehen Sie in das Verzeichnis `postgresql-7.3.4`, und führen folgende Befehle aus:

---

[3] Bezüglich der Postgres Installation ist Herrn Wehling von der Uni Köln aufgefallen, daß die Postgres 7.2 Versionen unter dem neuen Redhat 9 nicht kompilierbar sind.
Wenn man in

```
/install/postgres-7.2.4/src/backend/commands/copy.c
```

oben folgende Zeile einbaut, geht es:

```
#include <errno.h>
```

[4] Unter Suse Linux 8.x, 9.x und RedHat 8.x und Fedora ist der Postgres-Datenbankserver für internationale Umgebungen installiert und daher problemlos lauffähig. Siehe allerdings den Hinweis zur Sysconfig unter SuSE.

**Postgres 7.3.-8.x Installation "in short"**

```
./configure --enable-locale --enable-nls
make
make install
mkdir /usr/local/pgsql/data
chown postgres /usr/local/pgsql/data
```

Wenn Sie Postgres 7.2.x installieren, müssen beim `./configure` der Parameter `--enable-multibyte=LATIN1` gesetzt werden, in Postgres 7.3 oder höher ist dies defaultmäßig bereits eingebaut sind.

Damit sind die Schritte, die als `root` auszuführen sind, beendet. Wir wechseln nun zur Kennung postgres mit

`su - postgres`

Vor der Initalisierung des DBSPACE sollte die Sprachumgebung des Users postgres korrekt sein. Für die `bash` wird in den meisten Distributionen die Umgebung generell in der Datei `.bashrc` im Homeverzeichnis des Users `postgres` gesetzt; dort geben Sie den Pfad für das `data`-Verzeichnis an, und legen die Ausführprogramme von Postgres in den Datenpfad. Hier ein Beispiel für den Betrieb mit UTF-8:

**.bashrc bei Codierung mit UTF-8:**

```
...
export LANG=de_DE.utf8
#Zur Sicherheit für Postgres auch einzeln:
export LC_CTYPE='de_DE.utf8'
export LC_COLLATE='de_DE.utf8'
export LC_TIME='de_DE.utf8'
export LC_NUMERIC='de_DE.utf8'
export LC_MONETARY='de_DE.utf8'
export LC_MESSAGES='de_DE.utf8'

PATH=$PATH:/usr/local/pgsql/bin
export PGDATA=/usr/local/pgsql/data
export PGLIB=/usr/local/pgsql/lib
...
```

und hier ein Beispiel für ISO:

**.bashrc bei Codierung in ISO**

```
...
export LANG=de_DE@euro
#Zur Sicherheit für Postgres auch einzeln:
export LC_CTYPE='de_DE@euro'
export LC_COLLATE='de_DE@euro'
export LC_TIME='de_DE@euro'
export LC_NUMERIC='de_DE@euro'
export LC_MONETARY='de_DE@euro'
export LC_MESSAGES='de_DE@euro'

PATH=$PATH:/usr/local/pgsql/bin
export PGDATA=/usr/local/pgsql/data
export PGLIB=/usr/local/pgsql/lib
...
```

Wenn die Sprachumgebung stimmt, dann wird der DBSPACE vom User postgres initialisiert.

**Initialisierung des DBSPACE**

```
/usr/local/pgsql/bin/initdb -D $PGDATA
```

Durch `initdb` wird der DBSpace erzeugt. Wenn die Umgebung stimmt, dann wird Postgres für die deutsche Locale vorbereitet (Sortierung von Zeichen, Datums- und Währungsformate etc).

**Ausgabe von initdb**

```
/usr/local/pgsql/bin/initdb -D $PGDATA
The files belonging to this database system will be owned by user
"postgres".
This user must also own the server process.

The database cluster will be initialized with locale de_DE.utf8.
This locale setting will prevent the use of indexes for pattern matching
operations.  If that is a concern, rerun initdb with the collation order
set to "C".  For more information see the Administrator's Guide.
```

**Hinweis für SuSE-Anwender**

Wenn Sie Postgres als Binärpaket aus der Distribution von SuSE 8.x - 10.x verwenden, müssen Sie beachten, dass der DBSPACE beim ersten Start des Postmaster automatisch in `/var/lib/pgsql/data` angelegt wird. Wenn dabei die Umgebungsvariablen nicht auf die deutsche Locale gesetzt sind, wird ein amerikanischer Zeichensatz benutzt (Default-Einstellung). Bevor Sie also das Init-Script z.B. im Runlevel-Editor des `YAST` starten, sollten Sie mit dem Sysconfig-Editor (im Yast: `System->Editor für Sysconfig-Dateien`) die Variable `POSTGRES_LANG` (im Yast: Suche nach "POSTGRES") auf die deutsche Locale (`de_DE.utf8` oder `de_DE@euro` setzen. Außerdem sollten Sie dann die Variable
`POSTGRES_OPTIONS=-i`
setzen.
Noch ein Fallstrick in SuSE 9.1: Wegen eines Bugs im Yast funktioniert die Suche im Sysconfig-Editor nur im Textmodus.

Dann müssen Sie die ip-Nummer des Rechners mit dem SuperX-Webserver (sowie von allen anderen Clients, die direkt auf die Datenbank zugreifen sollen) in die Datei `/usr/local/pgsql/data/pg_hba.conf` eintragen. In der Datei `$PGDATA/pg_hba.conf` stehen die Verbindungsberechtigungen für der Server; hier müssen Sie mindestens dem User `superx` die Verbindungsrechte geben, z.B. mit folgender Zeile:

**Auszug aus pg_hba.conf**

```
host    all        all        127.0.0.1/32        trust
host    all        all        192.168.0.16/32     trust
```

Die obige Zeile gibt dem User `superx` Verbindungsrechte für alle Datenbanken auf dem lokalen Rechner `192.168.0.16`.

Bitte beachten Sie, dass die Standardvorgabe nach der Installation von Postgres die ist, dass alle User auf dem aktuellen Rechner mit dem Datenbankserver verbinden dürfen. Dies sollten Sie natürlich ändern.

Weitere Parameter werden in der Konfigurationsdatei `postgresql.conf` definiert; wichtig ist die Einstellung, dass Postgres einen TCP-IP-Socket öffnet (Parameter `tcpip_socket=true` bei Postgres 7.x, `listen_addresses=<<IP-Nr.>>` bei Postgres 8.0 oder höher) sowie der TCP-IP-Port (port = 5432 ist die Standardvorgabe). Die Anzahl der gleichzeitig offnenen Verbindungen muss kleiner sein als die Anzahl, die Sie für das SuperX-Servlet definieren. Weitere Details zur Einrichtung von Postgres-Runtime-Parametern

finden Sie im Admin-Handbuch der Postgres-Distribution. Außerdem sollen Sie beim Betriebssystem SuSE 9.1 oder höher den IPV6-Eintrag für "localhost" (::1) in /etc/hosts auskommentieren.

Danach wird der Datenbankserver gestartet mit dem Befehl postmaster.

```
/usr/local/pgsql/bin/postmaster -i -D /usr/local/pgsql/data
```

Wir empfehlen, die Ausgabe von dem Prozeß in eine Logdatei zu schreiben, z.B. nach /var/log/postgresql.log. Legen Sie diese Datei als User root an, und machen Sie dann den User postgres zum Eigentümer. Ein Beispielscript ist folgendes (im Kernmodul zu finden unter $SUPERX_DIR/db/install):

**pgsql_start.x**
**Ein Beispielscript zum Start von Postgres**

```
#!/bin/sh
PG_HOME=/usr/local/pgsql
export PG_HOME
PGDATA=$PG_HOME/data
export PGDATA
PGPORT=5432
export PGPORT
$PG_HOME/bin/pg_ctl -D $PGDATA -l /var/log/postgresql.log -o
-i start
```

Um zu testen, ob die Locale richtig ist, gehen Sie als User postgres in die Shell:

**Prüfen der Locale**

Öffnen Sie mit

```
psql template1
```

die Datenbank; dann geben Sie ein:

```
select 'aaa' union select 'bbb' union select 'äää' order by 1;
```

Bei richtiger Locale lautet die Ausgabe:

```
?column?
----------
 aaa
 äää
 bbb
(3 rows)
```

Im Verzeichnis $SUPERX_DIR/db/install befindet sich ein Shellscript check_sortierung_pg.x, das prüft, ob die aktuell in der Umgebung festgelegten Variablen zu korrekter Darstellung von Umlauten und Sortierung unter Postgres der gewünschte Ergebnis bringen. Das Script legt einen temporären DBSPACE an, führt darin einen Testselect aus und löscht den DBSPACE wieder, in der Logdatei check_sortierung.log steht dann das Ergebnis. In dem Script muss die Variable PG_HOME korrekt gesetzt sein, der Rest wird automatisch geprüft.

Dann erzeugen Sie den User superx für Postgres:

```
createuser superx
```

Dieser User muss Datenbanken erzeugen dürfen, braucht aber, wenn Sie als SuperUser bereits die Prozedursprache plpgsql in template1 installiert haben, kein Super-User sein bzw. bei Postgres 7.4 das Recht haben, andere User erzeugen zu dürfen [5]. Aus Sicherheitsgründen empfehlen wir, den User superx, der standardmäßig auch der User ist, mit der die Webapplikation auf die Datenbank zugreift, nicht zum Super-User zu machen.

Bei Änderungen der pg_hba.conf müssen Sie übrigens Postgres nicht neu starten, Sie können die Datei im laufenden Betrieb auch mit pg_ctl -D $PGDATA reload neu laden.

---

[5] In den 7.x-Postgres-Versionen ist dies offensichtlich ein Bug: Wenn ein User Datenbanken und Benutzer anlegen darf, dann wird er von Postgres als "Superuser" klassifiziert und darf deshalb auch Scriptsprachen installieren. In Postgres 8.x wurde dies korrigiert.

SuperX benötigt die Prozedursprache **plpgsql**. Wenn Sie als SuperUser die Prozedursprache installieren wollen, geben Sie in der Shell ein:

```
createlang plpgsql
```

Damit ist Postgres installiert und für die SuperX-Installation konfiguriert. Bei dieser Gelegenheit sollten Sie den Datenbankserver gleich als Dienst beim Systemstart einrichten.

#### *2.1.4.1.4.2Postgres-Zusätze installieren: pgcrypto*

Neben dem Kernsystem von Postgres bietet es sich an, die vielen Zusatzmodule von Postrges zu nutzen. Die Installation erfolgt aus den Quellen der Kerndistribution. Wir zeigen dies am Beispiel von **pgcrypto**, einem Paket zur Verschlüsselung, das wir für die Verschlüsselung von Passwörtern gebrauchen:

Nach dem `./configure` (s.o.) der gesamten Postgres-Quellen gehen Sie als root in das Verzeichnis `contrib/pgcrypto`

Geben Sie ein:

```
gmake all
gmake install
```

Es werden Bibliotheken in `/usr/local/pgsql/lib` erzeugt. Das SQL-Script zur Erzeugung der Crypo-Funktionen liegt in `/usr/local/pgsql/share/contrib/pgcrypto.sql`. Wenn Sie es in der SuperX-Datenbank installieren wollen, geben Sie dort ein:

```
psql superx < pgcrypto.sql
```

Wenn Sie es allen Datenbanken zur Verfügung stellen wollen, laden Sie die Funktionen nach template1:

```
psql template1 < pgcrypto.sql
```

#### *2.1.4.1.4.3Installation von Postgres unter Windows*

Für die Installation von Postgres unter Windows existiert seit Postgres 8.0 eine Möglichkeit, Postgres nativ zu betreiben. Dies empfehlen wir. Aus historischen Gründen haben wir auch den Betrieb von Postgres unter Cygwin dokumentiert.

Für den Betrieb von SuperX wird aber auf jeden Fall die Shell-Umgebung von Cygwin benötigt. Dies wird in einem dritten Abschnitt erläutert.

#### *2.1.4.1.4.4Native Windows-Version (nur PowerGres, Postgres 8.0 oder höher)*

Seit längerem gibt es eine kostenpflichtige Windows-Version von Postgres unter dem Namen PowerGres. Mit der Version 8.0 läuft auch das "normale" Postgres nativ (d.h. ohne die Unix-Emulation Cygwin) unter Windows, allerdings nur unter Win2000 und WinXP (nur XP Professional, nicht XP Home). Dies bietet erheblich mehr Komfort bei der Installation und Stabilität beim Betrieb. Für SuperX müssen Sie aber in jedem Fall `cygwin` installieren (s.u.), da die SuperX Scripte nur unter Unix / bash laufen.

Laden Sie die neueste Version von Postgres (Win) herunter.

- Installieren Sie als Administrator das msi-Paket, z.B. im Verzeichnis `C:\Programme\PostgreSQL\8.0-beta1`. Achten Sie darauf, daß alle Pakete installiert werden, auch **pgadmin III** (ältere pgadmins, odbc- oder jdbc-Treiber funktionieren nicht).

- Der User, der postgres startet, muss ein normaler User sein(z.B. "postgres"), kein Administrator; er muss vorher unter Windows angelegt sein. Er ist auch der Eigentümer der Datenbank **template1** (der Superuser).
- Postgres sollte als Dienst installiert werden
- Beim Anlegen des Datenbank-Cluster legen sie die deutsche Locale an, und als Zeichenformat LATIN1 (nicht unicode). Das Dateisystem muss NTFS sein.
- psql & co dürfen für den Betrieb von SuperX beim User nicht in den Windows-PATH gesetzt werden (z.B. `C:\Programme\PostgreSQL\8.0-x\bin`), stattdessen nehmen wir die Cygwin-Applikationen (s.u.).
- in `C:\Programme\PostgreSQL\8.0-x\data\postgresql.conf` muss man statt
  früher tcpip_socket = true
  den Parameter listen_adress ='IP-Adresse'
- In der Datei `pg_hba.conf` ist die Standardanmeldung anders als unter Unix auf md5 (nicht trust) gesetzt; wenn Sie nicht ständig das User-Passwort eingeben wollen, sollten Sie den entsprechenden Passus auf "trust" setzen.

Damit ist Postgres konfiguert, Sie können den Dienst jederzeit in der Computerverwaltung über das Applet "Dienste" neu starten. Normalerweise startet Postgres dann auch beim Systemstart automatisch.

##### *2.1.4.1.4.5Postgres unter Cygwin*

Neben der nativen Postgres-Installation (die wir empfehlen) gibt es auch die Möglichkeit, Postgres unter Cygwin zu betrieben. Insgesamt eignet sich eine unter Cygwin kompilierte Postgres-Installation unter nur für den Testbetrieb, denn bei der Sortierung werden Umlaute falsch eingeordnet und es wird sehr großzügig mit der Prozessorlast umgegangen: Wenn Postgres-Prozesse laufen, dann ist die Performance des Rechners für andere Anwendungen weitgehend gesperrt.

Aber auch bei der nativen Postgres-Installation unter Windows benötigen Sie für Postgres und SuperX unter Windows die UNIX-Shell-Emulation **cygwin**. Cygwin bietet rudimentäre UNIX-Funktionen wie z.B. die "bash", aber keine UNIX-typischen Dateirechte (z.B. Ausführungsrechte für User, Gruppen oder Andere).Außerdem unterstützt Cygwin (unseres Wissens) keine Locales, und unter Win98 haben wir keine stabile Installation hinbekommen. In den Mailinglisten wurden häufiger Probleme mit Win98 berichtet, unter WinME, Win2000 und Windows XP haben wir Cygwin erfolgreich getestet.

Das folgende Beispiel arbeitet mit Postgres 7.4.x. Postgres ist als Paket im Installer von Cygwin auswählbar.

Für die Installation muss man eine Windows-Kennung benutzen, die Rechte für "Standardbenutzer" reichen aus (es sei denn Cygwin soll als Dienst laufen). Außerdem: Wenn Sie planen, Daten bzw. entladene Datenbank-Exporte zwischen verschiedenen Rechnern hin- und herzuschieben, sollten Sie darauf achten, dass Sie immer die gleiche Kennung benutzen. Sie können z.b. `superx` nehmen. Die Windows Kennung, unter der man Cygwin installiert, wird nämlich nach Cygwin durchgereicht.

Vorgehen:

1) Die setup-Datei `setup.exe` der Unix-Emulation Cygwin von http://www.cygwin.com herunterladen und starten
   Dann je nach Belieben direkt aus dem Internet installieren oder zunächst herunterladen und dann `install from local directory` (alle Komponenten ausgewählt lassen) anklicken (wir empfehlen

letzteres Vorgehen, da das Online-Cygwin-Paket ständig aktualisiert wird).

2) Als Installationspfad sollten Sie unbedingt einen Pfad wählen, der keine Leerzeichen enthält, z.B. `c:\cygwin`).
3) Bei der Frage, für welchen User Cygwin installiert werden soll, wählen Sie "All users", und beim Standard-Dateiformat wählen Sie Unix.
4) Bei der Auswahl der Pakete sollten Sie wie folgt vorgehen: Bei den **Shells** mauss auf jeden Fall die **bash** ausgewählt sein. Zusätzlich zu den Defaults müssen lediglich **Base -> TextUtils, Database -> Postgres, Admin -> cron, net->openssh** und **Libs -> libint** und **libint1** manuell ausgewählt werden. Ein Mailprogramm (mutt, mail) sollte auch installiert werden. Wenn Sie Postgres selbst aus den Quellen installieren wollen, dann wählen Sie natürlich nicht Postgres aus. Danach einmal starten, das `home`-Verzeichnis wird angelegt
5) Das Cygwin-/bin Verzeichnis muss in der Umgebungsvariable PATH vor den Windows Programm-Verzeichnissen liegen, denn die `sort.exe` von Cygwin muss benutzt werden, nicht die von Windows. Prüfen Sie außerdem im Verzeichnis /bin, ob die `bash.exe` existiert - dies muss der Fall sein.
6) Wenn Sie Postgres nativ (d.h. mit dem Windows-Installer von Postgres ab Version 8.x) installiert haben, dann können Sie jetzt aufhören. Der folgende Teil gilt nur für Postgres unter Cygwin: IPC-Daemon starten
   ```
   ipc-daemon2 &
   ```
   Danach ist Postgres bereits installiert. Wenn Sie Postgres selbst aus den Quellen installieren, dann gehen Sie in das Verzeichnis mit den Quellen von postgresql. Die Installationsschritte entsprechen der Linux-Installation, außer dass Sie beim `configure` auch `--enable-odbc` eingeben sollten. Wenn entsprechende Fehlermeldungen erscheinen, müssen Sie noch dafür sorgen, dass (am Beispiel einer Installation von Cygwin in c:\cygwin) `C:\cygwin\usr\local\pgsql\lib\pq.dll` im PATH ist.
7) Nach der Installation Cygwin neu starten; danach muss unter cygwin ein User installiert werden. Geben Sie dazu ein
   ```
   mkpasswd -d | grep <<Windows-Username>> >> /etc/passwd
   ```
   Unter Win95/98/ME muss man das Passwort in `/etc/passwd` noch verschlüsseln; ersetzen Sie den Passus "use crypt" durch die Ausgabe von dem Befehl
   ```
   crypt <<Ihr Passwort>>
   ```
8) Zur Initialisierung von Postgres folgendes eingeben:

```
ipc-daemon2 &
initdb -D /usr/local/pgsql/data
```

in `/usr/local/pgsql/data/postgresql.conf  #tcpip_socket=false`

```
# wegnehmen und auf true setzen
```

Zum Start des Postmaster eine Batchdatei z.B. `pgsql_start.x` anlegen mit dem Inhalt:

**pgsql_start.x**

```
#! /bin/sh
ipc-daemon2 &
pg_ctl -D /usr/local/pgsql/data -l /var/log/postgres.log -o -i start
```

Danach gibt man ein:

```
chmod +x pgsql_start.x
./pgsql_start.x
```

Der Postmaster startet dann, und die Logdatei `/var/log/postgres.log` wird gefüllt.

Den erfolgreichen Start von Postgres kann man prüfen, indem man `psql template1` eingibt.

Den postmaster beendet man wie unter UNIX mit
`pg_ctl stop –D /usr/local/pgsql/data`

Die Installation des Kernmoduls kann danach vorgenommen werden; bei der Umgebungsvariable `JAVA_HOME` müssen Sie die Windows-Installation von Java verwenden `(/cgydrive/<<Windows-Laufwerk>>/<<Pfad zum JDK>>).`

Noch ein kleiner Hinweis: Wenn Sie sich von entfernten Rechnern auf dem Cygwin-Server anmelden wollen, müssen Sie den ssh-Daemon installieren (s.u.).

##### *2.1.4.1.4.6Cygwin für SuperX*

Für die Modulscripte von SuperX wird die leistungsfähige Scripting-Umgebung Cygwin benötigt (unter Windows / DOS gibt es nichts Vergleichbares!). Gleichzeitig bleiben dadurch SuperX-Distributionen plattformübergreifend, durch geringe Anpassungen erreichen wir, dass Scripte unter Unix auch unter Cygwin laufen. Allerdings können Sie Cygwin nur in Verbindung mit Postgres nutzen, nicht mit Informix, weil der Informix-Client dbaccess (nach unserem Wissen) nicht unter Cygwin läuft.

Die folgenden Ausfürhungen gelten also nur für Postgres-Anwender: Sie installieren also zunächst wie oben beschrieben Cygwin und Postgres, allerdings <u>ohne</u> das Paket IPC-Daemon zu installieren. Bei nativem Windows-Betrieb muss der oben bei Cygwin genannte `cygipc`-Dienst nicht installiert und gestartet werden. Im Folgenden ein paar Anpassungen für die Bash unter Cygwin.

Beachten Sie, dass in der Konfigurationsdatei `$SUPERX_DIR/db/bin/SQL_ENV` die Umgebungsvariable `PGHOST` gesetzt sein muss, und dass der Pfad für die Binaries von Postgres angepasst werden muss.

**Auszug aus der SQL_ENV für Cygwin und Postgres (nativ)**

```
case $SX_CLIENT in

psql)
        export PGDATESTYLE=German
        O_DESCR=$SUPERX_DIR/db/conf/unldescr_postgres_copy.xml
        export O_DESCR
        PGPORT=5432
        export PGPORT
        #Bei Betrieb von Postgres unter Win muss für psql
        #unter cygwin die Umgebungsvariable PG_HOST gesetzt sein
        #Sonst versucht er eine Socket Connection
        PGHOST=localhost
        export PGHOST
        #Prüfen ob der PATH erweitert werden muss
        PGPATH=/bin
        case $PATH in
        *$PGPATH*)
                ;;
        *)
                export PATH=$PATH:$PGPATH
                #echo "PATH erweitert"
                ;;
        esac
;;
```

Wenn Sie Cygwin und Postgres-Windows auf einem Rechner nutzen, müssen sie darauf achten, dass beim Öffnen der Cygwin-Shell in der Umgebungsvariable "PATH" auf jedne Fall der Pfad zum Cygwin-psql (normalerweise in `/bin`) vor dem Eintrag zum DOS-psql (nomalerweise unter

C:\Programme\Postgresql<<Version>>\bin) liegt, denn die SuperX-ETL-Scripte können mit dem DOS-psql nicht arbeiten.

Noch ein Hinweis für ältere SuperX-Versionen (2.x): Der alte jdbc-Treiber pgjdbc2.jar im Verzeichnis %SUPERX_DIR%\webserver\tomcat\webapps\superx\WEB-INF\lib muss gelöscht und durch den mitgelieferten Treiber pg74.214.jdbc3.jar ersetzt werden. Entsprechende Verweise in der Datei $SUPERX_DIR/db/bin/SQL_ENV (Umgebungsvariable JDBC_CLASSPATH) müssen entsprechend geändert werden.

Wenn Sie auch einen SSH-Zugriff aus dem Rechner ermöglichen wollen (dies empfehlen wir u.a. wg. der Dateiübertragung mittels rsync), müssen Sie den SSH-Dämon unter Cygwin starten. Dazu müssen Sie zunächst eine Cygwin-Shell öffnen, und dort eingeben:

```
ssh-host-config
```

Es werden einige Dateien generiert, und außerdem werden ein paar Einstellungen abgefragt. Bei dem Fragen zum Account für den SSH-Daemon antworten Sie mit "no", d.h. der aktuelle Cygwin User startet den Dämon (dieser ist ohnehin kein Admin-User). In diesem Falle lässt sich cygwin aber nicht als Dienst einrichten.

Danach startenSie den SSH-Server mit

```
/usr/sbin/sshd
```

Danach können Sie sich mit Putty auf dem Server einloggen.

#### *2.1.4.1.4.7Postgres-Performance-Tipps*

Der Optimierer unter Postgres läßt sich uber die Kommandozeile mit

```
vacuumdb --analyze --verbose -f -d $DBNAME
```

starten und hilft bei regelmäßiger Anwendung, deshalb empfehlen wir, diesen Befehl als Cronjob jede Nacht oder einmal pro Woche auszuführen.

Wichtige Parameter sind die "Shared buffers" und der "Effective cache size". **Shared buffers** meinen nicht das gesamte zur Verfügung stehende RAM (das verwaltet das System), sondern den Arbeitsspeicher, der Postgres zum Zwischenspeichern benutzt, bevor Abfragen an den Kernel geschickt werden. Der Wert sollte nicht zu hoch gewählt werden, weil dann die Performance nachlässt. Faustregel: 6-15% des physischen RAM, man sollte aber auch in der Praxis viel probieren.  Generell sollte man auf Datenbankservern mind. die Hälfte des verfügbaren physischen Rams für Postgres reservieren.

Beim Start des Postmasters lassen sich die "Shared buffers" zuweisen mit der Option

```
postmaster –o "–B 128"
```

Dabei wird das Shared Memory von (standardmäßig) 64*8192 Bytes auf 128*8192 Bytes erhöht.  Man kann den Parameter aber auch in der postgresql.conf setzen.

**Beispielkonfiguration Postgres-RAM bei DB-Server mit 1 GB RAM unter Suse Linux**

in der Datei /etc/init.d/boot.local geben Sie ein:

```
echo 536870912 > /proc/sys/kernel/shmmax #512 MB RAM für PG
echo 2097152 > /proc/sys/kernel/shmall
echo 2 > /proc/sys/vm/overcommit_memory
```

Die Parameter lassen sich auch zur Laufzeit aus einer root-Shell setzen. Danach ersetzen Sie in der postgresql.conf die folgenden Parameter:

```
max_connections = 500
shared_buffers = 16384
max_fsm_pages = 50000
checkpoint_segments = 12
effective_cache_size = 32000
```

Danach starten Sie Postgres neu.

Die checkpoint segments sollen Sie erhöhen, wenn Sie in den Postgres-Logs folgende Meldung bekommen:

```
LOG:  Checkpoints passieren zu oft (alle xx Sekunden)
TIPP:  Erhöhen Sie eventuell den Konfigurationsparameter "checkpoint_segments".
```

In der Postgres-Auslieferung sind checkpoint segments=3 vorgegeben, bei großen Anwendungen sollten Sie großzügig erhöhen, z.B. 24.

Effective Cache Size sollte als Faustregel 25% des physischen RAM betragen.

Diese und weitere Perfomance-Tipps für das jeweilige Betriebssystem finden Sie im PostgreSQL Administrator's Guide im Abschnitt "Run-Time Configuration".

Leider lassen sich Transaktionen für Postgres nicht abschalten, für ein (passives) Berichtssystem wie SuperX wären Transaktionen unbedeutend.

Autovacuum wurde mit Postgres 8 eingeführt. Für SuperX empfehlen wir dies nicht, denn das Vacuum wird in der Laderoutine ohnehin jede Nacht ausgeführt, und Autovacuum-Prozesse stören die Laderoutine, teilweise empfindlich.

#### 2.1.4.1.5Datenbankverbindung über einen eingeschränkten User für mehr Sicherheit

Zur Erhöhung der Sicherheit ist es möglich, dass die Datenbankverbindung von Tomcat zur Datenbank mit einem eingeschränkten User durchgeführt wird. Richten Sie dazu einen entsprechenden User in Ihrer Datenbank ein und geben Sie diesen beim Propadmin bei eingeschränkter User an. Der erste im Propadmin auszufüllende User muss weiterhin umfassende Rechte auf alle Tabellen haben, weil er auch bei Modulinstallationen/-updates verwendet wird. Das Minimum, was der eingeschränkte User haben muss sind select-Rechte auf alle Tabellen, insert-Rechte auf die Tabelle protokoll und update-Rechte auf userinfo.

Sobald Sie Ihre db.properties mit dem Propadmin bearbeitet haben, können Sie praktisch die Minimal nötigen Rechte vergeben, in dem Sie einmal das Skript

sx_restrictedconnmanager.x false aufrufen.

Nach einem Tomcat-Neustart findet sich in der catalina.out nach "Aufbau von Datenbank-Connection-Pool (..) .. OK" ein Hinweis:

eingeschränkter Datenbankuser für Verbindung: true|false

Wenn Sie Funktionen wie User/Gruppe/Maske einrichten/löschen etc. im XML-Frontend benutzen wollen, müssen zusätzliche Kernmodultabellen freigeschaltet werden:

```
-protokoll
-userinfo
-groupinfo
-user_institution
-user_sachgeb_bez
-user_masken_bez
-group_sachgeb_bez
-group_masken_bez
-user_group_bez
-user_pw
-user_sichten
-user_sichtarten
-group-sichten
-group_sichtarten
-felderinfo
-maskeninfo
-maske_system_bez
-masken_felder_bez
-sachgeb_maske_bez
-organigramm
-themenbaum
```

Am einfachsten können Sie dies erledigen, indem Sie das Skript `sx_restrictedconnmanager.x true` aufrufen.

#### 2.1.4.1.6Automatischer Start des Datenbankservers als Dienst

Nach erfolgreicher Installation des Datenbankservers muss der Server als Dienst eingerichtet werden. Wir haben das Vorgehen für die Betriebssysteme RedHat 8.0 und SuSE Linux 7.x-8.x beschrieben (für Debian ebenfalls, aber diese Scripte haben wir noch nicht getestet).

##### *2.1.4.1.6.1Einrichtung der Dienste*

Im Kernmodul befinden sich unter `/home/superx/db/etc` die Vorlagen für den DB-Server. Die Ordnerstruktur entspricht dem Linux-Rechners auf oberster Ebene. Kopieren Sie die Dateien als root in die entsprechenden Verzeichnisse, z.B. bei Redhat-Linux

`$SUPERX_DIR/db/etc/init.d/superx_db.redhat`

nach

/etc/init.d/superx_db

Ebenso verfahren Sie mit den Dateien in $SUPERX_DIR/db/etc/sysconfig.

Dann machen Sie die User informix / postgres zu Eigentümern der Dateien.

Die Variablen, die ggf. angepasst werden müssen, sind

SUPERX_USER

JAVA_HOME und andere Variablen aus $SUPERX_DIR/db/bin/SQL_ENV

(wenn Sie SuperX in einem anderen Verzeichnis als /home/$SUPERX_USER installiert haben, müssen Sie die Pfade zu TOMCAT_START und TOMCAT_STOP entsprechend anpassen).

Dann erzeugen Sie als root die leere Datei

/var/log/superx.log

und machen den User superx zum Eigentümer

chown superx:users /var/log/superx.log

Analog verfahren Sie mit

- /var/log/informix und machen den user informix zum Eigentümer
  bzw.
- /var/log/postgres und machen den user postgres zum Eigentümer
-

Dann

- kopieren Sie die Datei $SUPERX_DIR/db/etc/home_informix/start.sh in das Homeverzeichnis von Informix,
  und machen den user informix zum Eigentümer
  bzw.
- kopieren Sie die Datei $SUPERX_DIR/db/etc/home_postgres/start.sh in das Homeverzeichnis von postgres,
  und machen den user postgres zum Eigentümer.

Kontrollieren Sie, ob die Datei start.sh Ausführungsrechte besitzt.

#### *2.1.4.1.6.2Aktivierung der Dienste*

Zur Aktivierung der Dienste für den Runlevel 3 führen Sie jeweils folgende Schritte durch; erzeugen Sie einen symbolischen Link für das Script superx-db im Runlevel 3 und 5

**Redhat/Mandrake:**

```
ln -s /etc/rc.d/init.d/superx_db /etc/rc.d/rc3.d/S90superx_db
ln -s /etc/rc.d/init.d/superx_db /etc/rc.d/rc5.d/S90superx_db
ln -s /etc/rc.d/init.d/superx_db /etc/rc.d/rc3.d/K90superx_db
ln -s /etc/rc.d/init.d/superx_db /etc/rc.d/rc5.d/K90superx_db
```

**SuSE 8.x:**

```
ln -s /etc/init.d/superx_db /etc/init.d/rc3.d/S98superx_db
ln -s /etc/init.d/superx_db /etc/init.d/rc5.d/S98superx_db
ln -s /etc/init.d/superx_db /etc/init.d/rc3.d/K98superx_db
ln -s /etc/init.d/superx_db /etc/init.d/rc5.d/K98superx_db
```

**Debian,LSB:**

```
ln -s /etc/init.d/rc/superx_db /etc/init.d/rc3.d/S98superx_db
ln -s /etc/init.d/rc/superx_db /etc/init.d/rc5.d/S98superx_db
ln -s /etc/init.d/rc/superx_db /etc/init.d/rc3.d/K98superx_db
ln -s /etc/init.d/rc/superx_db /etc/init.d/rc5.d/K98superx_db
```

Danach können Sie als root testen, ob die Scripte laufen, indem Sie als root

`/etc/init.d/superx_db start`

zum Starten der Datenbank ausführen, und sowie

`/etc/init.d/superx_db stop`

zum Stoppen der Datenbank.

Etwaige Fehlermeldungen stehen in Logdatei `/var/log/informix.log`, `postgres.log` bzw. `superx.log`

### 2.1.4.2 Einspielen des Kernmoduls der SuperX-Datenbank

Für die Installation haben wir eine Kurzanleitung vorbereitet. Das Kernmodul der Datenbank liegt exportiert vor und kann in das DBMS übernommen werden. Die nachfolgenden Installationschritte gehen davon aus, daß Sie keinen speziellen DBSpace für SuperX vorgesehen haben.

Das Installationsscript für die Datenbank befindet sich im Verzeichnis

`$SUPERX_DIR/db/install/kernmodul_erzeugen.x <<ggf. mit Name des DBSpace>>`

Das Script läuft nur, wenn die Parameter in der Datei `$SUPERX_DIR/db/bin/SQL_ENV` stimmen. Bei erfolgreichem Ablauf kommt eine Erfolgmeldung, im Falle eines Fehlers wird die Fehler-Logdatet `create.log` angezeigt. Wenn ein Fehler auftritt, müssen sie die Datenbank vor einem erneuten Ablauf des Scriptes droppen.

> Wenn Sie bei der Postgres Installation nicht den Parameter `--enable-multibyte=LATIN1` angegeben haben müssen Sie eventuell in dem Script `kernmodul_erzeugen.x` die Variable `ENCODING` auf `LATIN1` setzen.

Danach können Sie mit `dbaccess superx` (unter Informix) bzw. `psql superx` (unter Postgres) testen, ob die Datenbank verfügbar ist.

Schließlich sollten Sie die Tabelle `hochschulinfo` anpassen und die Daten Ihrer Hochschule dort eingeben, insbesondere die Hochschulnummer (apnr-Wert in `cifx` mit key=36).

### 2.1.4.3 Update und Sichern der Datenbank

Vor dem Start der Update-Scripte sollte immer eine Sicherung der Datenbank erfolgen. Für Backups ist es notwendig, die Datenbank regelmäßig zu exportieren. Beide Datenbanken bieten entsprechende Werkzeuge.Es bietet sich an, einen cronjob einzurichten, der zuerst das Backup vornimmt, und dann die einzelnen Module nacheinander aktualisiert.

Ein Beispiel-Eintrag der crontab des users superx liegt in `$SUPERX_DIR/db/module/crontab.sam`. Ein Beispiel-Update-Script liegt in `$SUPERX_DIR/db/module/update.x.sam` . Der Eintrag in der crontab, der das Script werktags um 18:00 Uhr startet, sähe dann wie folgt aus:

**Beispieleintrag in der crontab des users superx**

```
# Täglicher SuperX-Update um 18 Uhr
#
00 18 * * 1-5  /home/superx/db/module/update.x
>>/home/superx/db/module/update.log 2>&1
```

Ein Beispielinhalt für das Script `update.x` ist Teil des Kernmoduls:

**Beispiel-Updatescript für SuperX: update.x (Auszug)**

```
#!/bin/sh
#This is the central update script for SuperX.
. /home/superx/db/bin/SQL_ENV
LOG=$SUPERX_DIR/db/module/superx_update.log
#Stop Tomcat
$SUPERX_DIR/webserver/tomcat/bin/shutdown.sh  >$LOG 2>&1
#Dump Database
$SUPERX_DIR/db/install/dump_it.x >>$LOG 2>&1
#Now the Modules are updated:
$SOS_PFAD/sos_update.x >>$LOG 2>&1
$COB_PFAD/cob_update.x >>$LOG 2>&1
```

Ein Beispielscript, das die Datenbank sichert, liegt in `$SUPERX_DIR/db/install/dump_it.x`. Es erzeugt den Dump im Verzeichnis `$SUPERX_DIR/db/install`, prüft die erfolgreiche Sicherung und verschickt ggf. eine Fehler-Mail. Wenn Sie das Script in einem Cronjob betreiben wollen, müssen Sie als ersten Parameter `$SUPERX_DIR` übergeben.

Die Rücksicherung einer Datenbank ist mit dem Script `$SUPERX_DIR/db/install/restore_it.x` möglich.

#### 2.1.4.3.1 Ein Dump unter Informix

Die Datenbank lässt sich mit dem Kommando `dbexport –o <Pfad> superx` exportieren und sichern. Beachten Sie aber, dass durch das Servlet eine (oder mehrere) Verbindungen zur Datenbank geöffnet ist. Deshalb muss das Servlet beendet werden oder die Datenbank muss vom User Informix einmal auf `quiescent` und dann wieder auf `online` gesetzt werden, damit eventuell noch ablaufende SuperX-Prozesse beendet werden.

#### 2.1.4.3.2 Ein Dump unter Postgres

Postgres lässt sich auch im laufenden Betrieb sichern.

In unserem Dump-Script wird der Dump mit dem Parameter "--inserts" versehen. Dies ist eine sehr vorsichtige Einstellung, aber der Dump ist dadurch maximal kompatible zu verschiedenen Postgres-Versionen, außerdem tauchen keine Probleme mit Umbrüchen in langen Textfeldern auf.

Wenn Ihnen die resultierenden Dumps zu groß sind, können Sie in einem eigenen Dump auf die Inserts verzichten,z.B. mit

```
pg_dump -f superx.sql superx
```

Noch kompakter ist der Dump als Binärfile mit dem Parameter --format=c:

```
pg_dump -f  $DBNAME.sql --format=c $DBNAME
```

### 2.1.4.4Anpassung der DB-Parameter für Clientanwendungen

Zunächst ist es wichtig, eine Verbindung vom Webserver zum Datenbankserver zu bekommen. Dazu gibt es verschiedene Werkzeuge.

#### 2.1.4.4.1Unter WIN32 auf den Informix-Server zugreifen: iLogin

Wenn Sie Tomcat und den Webserver auf einem WIN32-Rechner betreiben wollen, dann ist es sinnvoll, zunächst die Datenbankverbindung zu überprüfen. Um von Windows-Rechnern auf Informix-Datenbanken zugreifen zu können, muss man dem Rechner den Service bekannt machen. Dazu muss man in der Datei `winnt\system32\drivers\etc\services`[6] den Port für den Service angeben, z.B. die Zeile

```
superx_server      1542/tcp
```

hinzufügen.

Nun können Sie Parameter für den Zugang von WIN32-Rechnern auf den Datenbankserver überprüfen. Der beste Weg dafür ist das Werkzeug iLogin, das von Informix in den Client-SDKs mitgeliefert wird. Die folgende Abbildung zeigt ein Beispiel für die Parameter beim iLogin

Die Parameter sind oben bereits erläutert. Ein erfolgreicher iLogin ist Voraussetzung für das weitere Vorgehen!



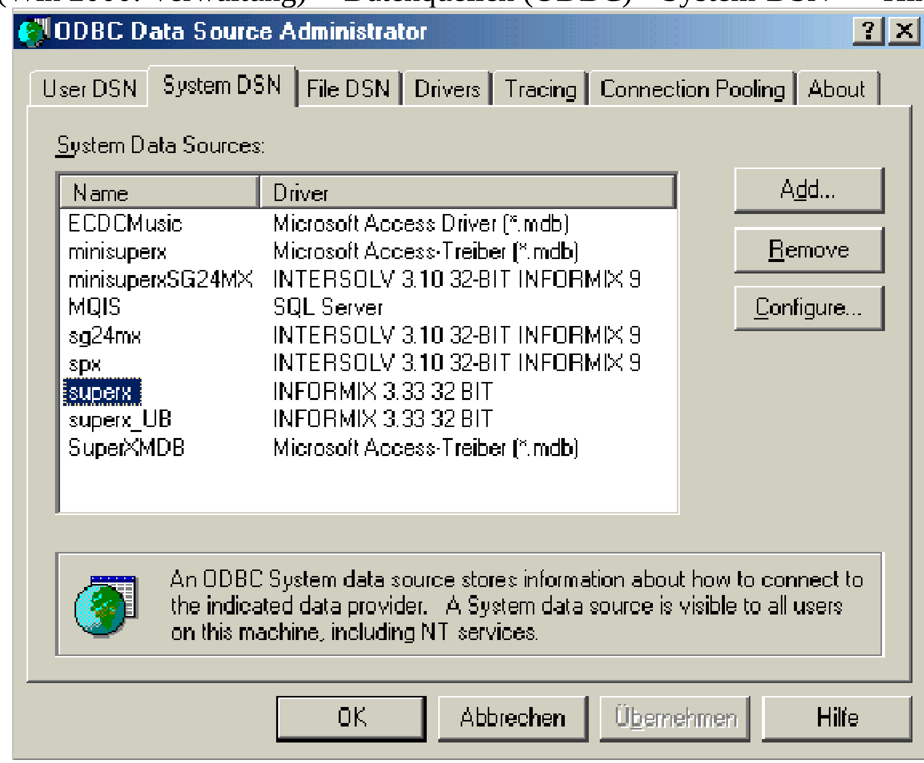
[6] Unter Win 98 / Me befindet sich diese Datei im Verzeichnis c:\windows

### *2.1.4.4.1.1SuperX (Informix) unter Win32 als ODBC-Datenquelle einrichten*

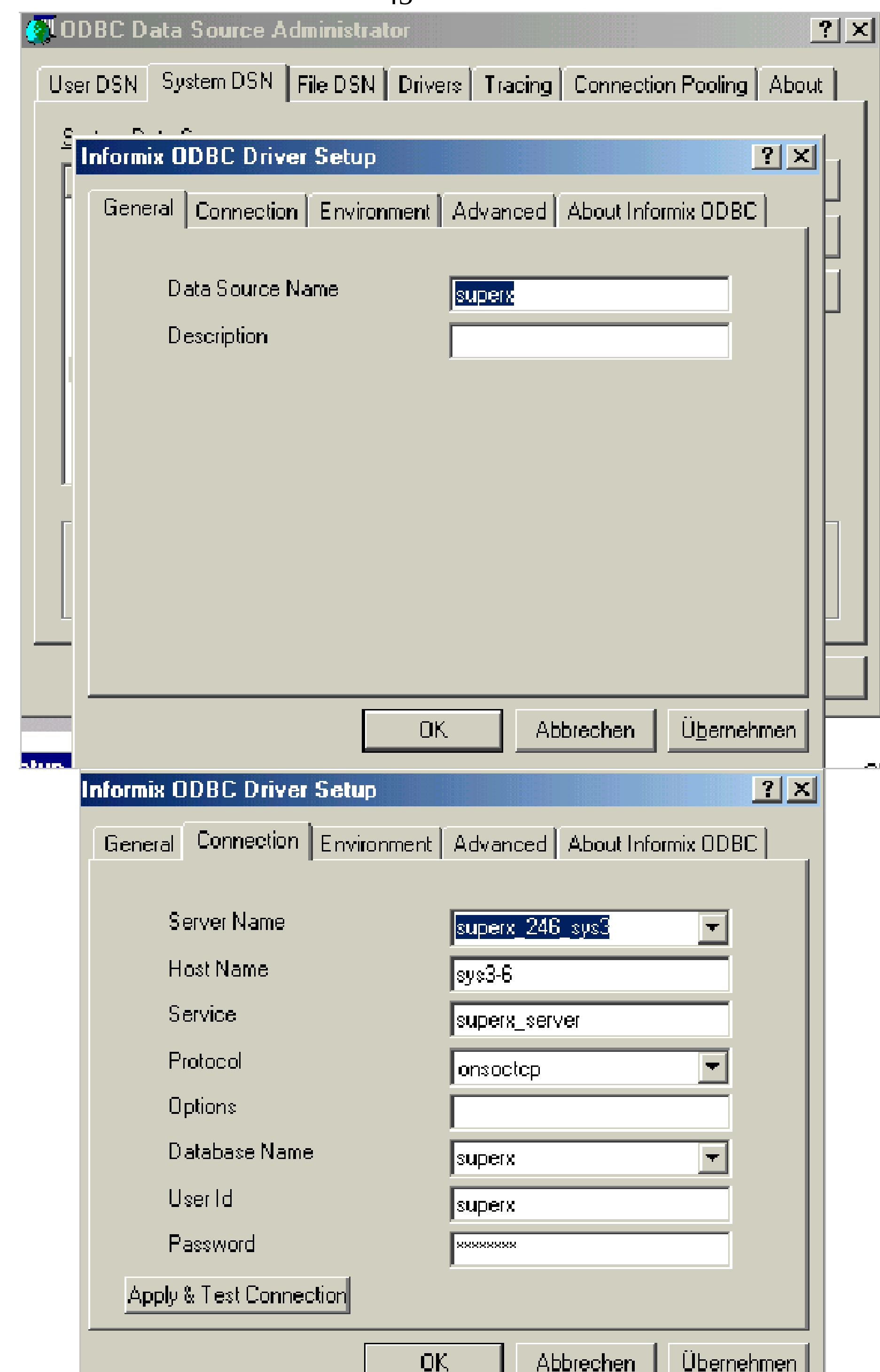
Für den regulären SuperX-Betrieb ist dieser Schritt nicht unbedingt erforderlich. Wenn Sie allerdings unter Win32 direkt auf die Datenbank zugreifen möchten, z.B. um Microsoft Access als Frontend einzusetzen, müssen Sie SuperX als ODBC-Quelle einrichten. Für die Informix-Datenbank gibt es eigene Treiber für den ODBC-Zugriff (für IDS 7.31 gibt es Intersolv 3.10 oder 3.11). Diesen Treiber muss man sich zunächst von www.informix.com besorgen. Meist sind die Treiber Teile des Informix Client SDK; für den reinen ODBC-Zugriff reicht es vollkommen aus, bei der Installation *Custom* zu wählen und nur den ODBC-Treiber zu installieren.

Zur Installation:

Systemsteuerung -> (Win 2000: Verwaltung)-> Datenquellen (ODBC)->System-DSN -> Hinzufügen




Für IDS 7.31 kann man als ODBC-Treiber z.B.den Intersolv 3.11-Treiber wählen. Dieser befindet sich im Informix Client SDK 2.40 (der 3.10-Treiber geht auch, der ist im Informix Client SDK 2.02). Für den IDS 9.21 benötigt man den Treiber Informix 3.33, der Teil des Client SDK 2.60 ist. Version 3.34 läuft ebenfalls.




Der Datenquellen-Name ist superx.

Als Datenbank-Name die SuperX-Datenbank angeben. Für die Verbindung die rechten Parameter eingeben (Achtung: Beispielangaben für Duisburg); wichtig sind der Hostname, der Service-Name (s.u.) und der Server.

Get-DB-List from Informix kann man deaktivieren. Manche ODBC-Treiber erlauben es in den erweiterten Optionen (Environment), eine DB-Locale zu definieren; wir empfehlen, diese auf Use Server Database Locale zu setzen.

Vorsicht mit ODBC

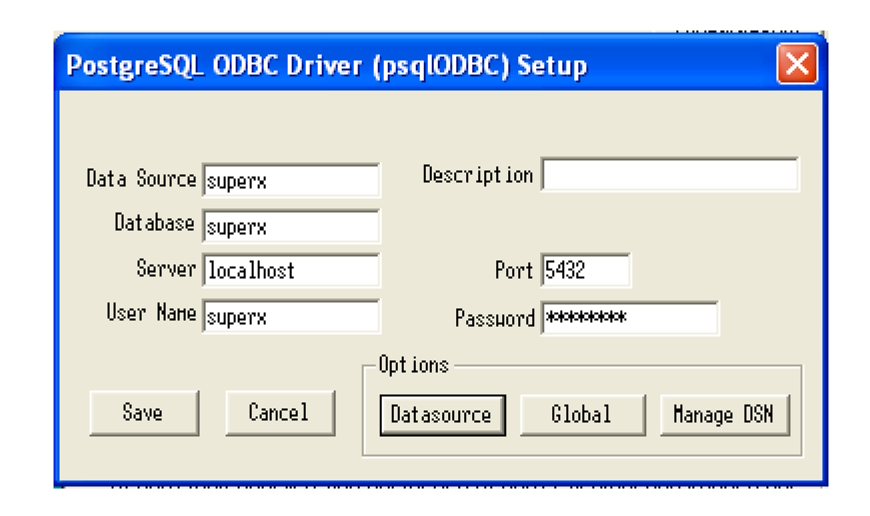
Der Informix-ODBC-Treiber 3.8 darf auf keinen Fall benutzt werden, er ist extrem fehlerhaft und kann zum Datenverlust führen.

Beachten Sie ggf., dass diese Installation unter NT/ Win 2000 kennungsabhängig ist.

#### 2.1.4.4.2 Einrichtung des ODBC-Treibers für den Postgres-Server

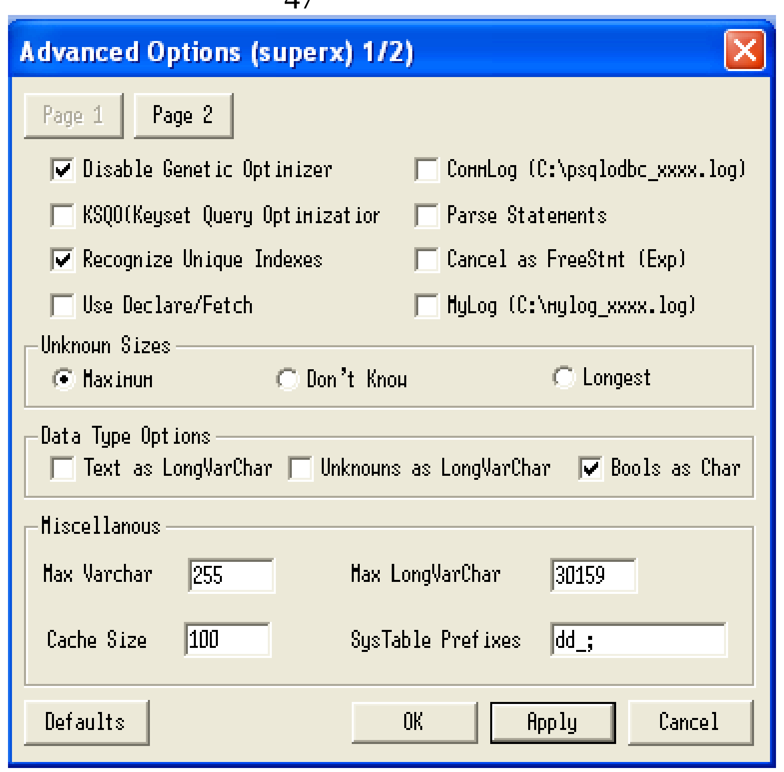
Der ODBC-Treiber für Postgres ist vom Postgres-Projekt verfügbar (www.postgresql.org). Er ist in der 8.0-Distribution von Postgres bereits enthalten. Der Treiber lässt sich leicht installieren, indem Sie in der Systemsteuerung über Verwaltung -> ODBC-Datenquellen eine Datenquelle einrichten, z.B. mit dem Namen superx.

Der Datenquellen-Name ist superx, der Datenbankname ebenfalls. Bei Server geben Sie den Hostnamen oder die IP-Nummer ein, und rechts den Port. Die Kennung ist hier z.B. superx.



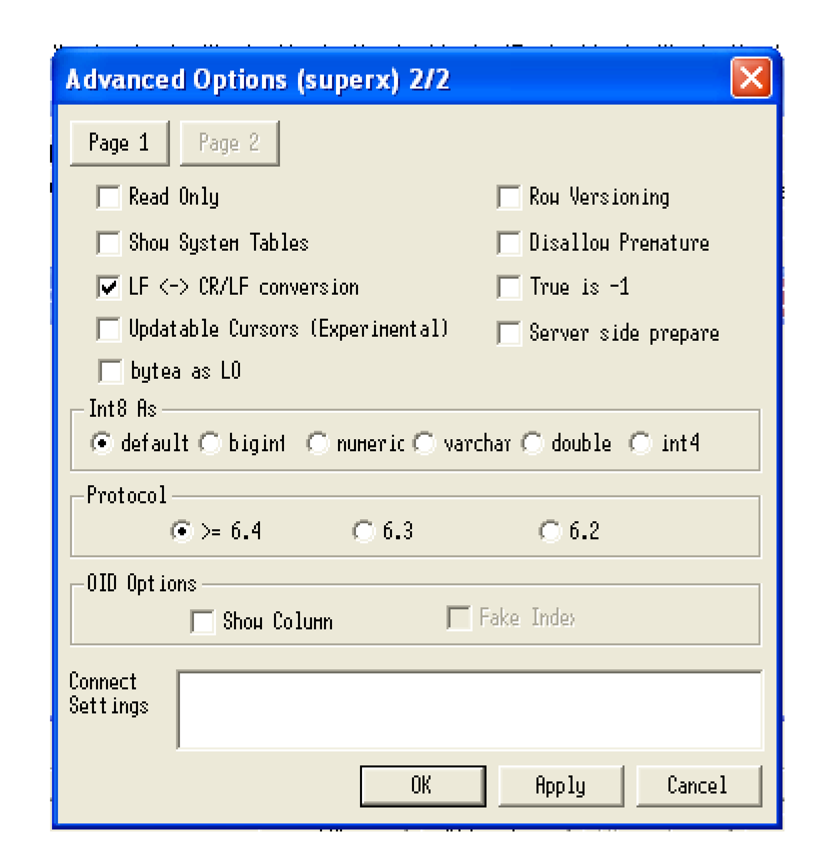

Im Dialog "Options-> Datasource" müssen einige Einstellungen vorgenommen werden:

Setzen Sie die Data Type Options wie rechts angezeigt. Das Kreuz bei `Bools as Char` ist notwendig, weil Access oder andere Frontends sich mit Postgres bei Binären Datentypen nicht vertragen. Boolean-Felder werden mit "1" oder "0" codiert.
Bei Max Varchar geben Sie 255 ein (sonst macht Access aus allen VARCHAR(255)-Feldern Memo-Felder), und Max LongVarchar mindestens 30.000. Der Rest ist ok.




Auf der zweiten Seite sind die Defaults korrekt.

Die Linefeed-Umsetzung ist wegen der Scripte in SuperX-Textfeldern notwendig.

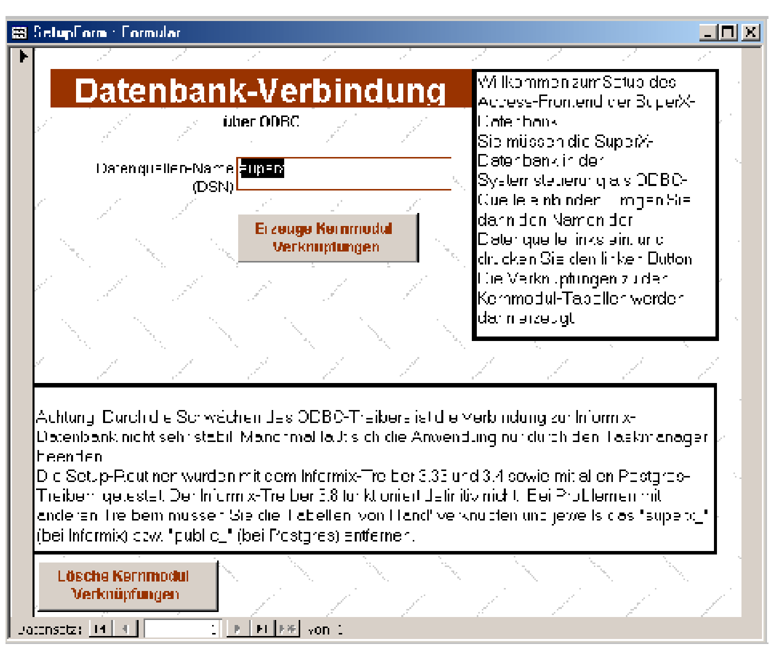



#### 2.1.4.4.3Anbindung des Access-Frontends an die ODBC-Quelle

Wenn Sie die SuperX-Datenbank als ODBC-Quelle unter dem Namen superx eingerichtet haben, dann können Sie das im SuperX-Clientpaket unter `$SUPERX_DIR/tools/access/superx_frontend_sam.mdb` enthaltene Access-2000-Frontend benutzen. Bei der Datei handelt es sich um ein Muster, vor dem ersten Gebrauch kopieren Sie sie bitte nach `$SUPERX_DIR/db/superx_frontend.mdb` und arbeiten Sie nur mit letzterer Datei - so können Sie sichergehen, dass Ihr Access-Frontend nicht bei einem SuperX-Update überschrieben wird.

Beim ersten Öffnen der Datei sind die Tabellen noch nicht verknüpft. Sie müssen zunächst Das Formular *Setup* aufrufen, den Namen der ODBC-Quelle (s.o.) eintragen, und "Erzeuge Kernmodul-Verknüpfungen" drücken. Wenn der Informix-Treiber dies unterstützt, sollte vorher die Option "Passwort speichern" aktiviert werden, ansonsten muss man für jede Tabellenverknüpfung das Passwort eingeben.

Die Datenquelle wird eingegeben, und die Kernmodul-Tabellen können so verknüpft werden. Die Verknüpfungen haben nach Access-Voreinstellung den Namen "superx_tabellenname" und werden automatisch umbenannt zu "tabellenname".




Falls der Setup so nicht funktioniert, müssen die Tabellen "von Hand" verknüpft und umbenannt werden. Die Funktionalität des Access-Frontends ist dadurch nicht beeinträchtigt. Bei Tabellen ohne Primary Key muss allerdings ein eindeutiger Datensatzbezeichner angegeben werden, sonst ist die Tabelle schreibgeschützt.

Bitte beachten Sie, dass die ODBC-Treiber von Informix recht instabil sind und die Anwendung sich manchmal nur durch den Taskmanager beenden läßt. Wir mussten bei Access 2000 und 2002 feststellen, dass einige Formulare nach einiger Zeit nicht mehr geschlossen werden können, und Visual-Basic-Routinen mit der ominösen Fehlermeldung "Dieser Vorgang wird den aktuellen Code in den Unterbrechungsmodus zurücksetzen" beenden. Dieser Fehler ist bei Microsoft dokumentiert, aber die vorgeschlagene Lösung hat bei uns nicht funktioniert[7]. Eine funktionierende Lösung fanden wir in Access-Foren[8].

[7] http://support.microsoft.com/default.aspx?scid=kb;DE;304548

[8] Ändern Sie mit regedit folgenden Schlüssel:
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Jet\VERSION\Engines\ODBC
Setzen Sie dort den (dezimalen) Wert ConnectionTimeout (z.B. 600) auf 0
Siehe http://www.ms-office-forum.net/forum/archivethread-111477.html oder
http://www.ms-office-forum.net/forum/showthread.php?s=&postid=438543#post438543

### 2.1.4.4.4Anpassen der Datenbankparameter für das SuperX-Servlet

Wenn Sie die Verfügbarkeit des Datenbankservers getestet haben (z.B. über das Utility iLogin von Informix), dann müssen die Datenbankparameter in die Datei

`$SUPERX_DIR/webserver/tomcat/webapps/superx/web-inf/db.properties`

übertragen werden, damit das Servlet eine Verbindung zur Datenbank herstellen kann. Ein Muster für Informix und eines für Postgres ist bereits im Kernmodul enthalten. Kopieren Sie die Datei `db-informix.properties` bzw. `db-postgres.properties` nach `db.properties`. Das voreingestellte Passwort lautet hier "anfang12".

Zur Erstellung und ggfs. Änderung dieser Datei gibt es ein Tool: propadmin.x. Das Shellscript liest aus der Umgebungsvariable `$DB_PROPERTIES` (oder über den ersten Parameter) den Speicherort der `db.properties` ein; in der Regel muss das die obige Position sein, damit das Servlet die Datei findet. Ausnahmen gibt es nur, wenn SuperX über den jdbc-Client auf eine andere Datenbank zugreifen soll.

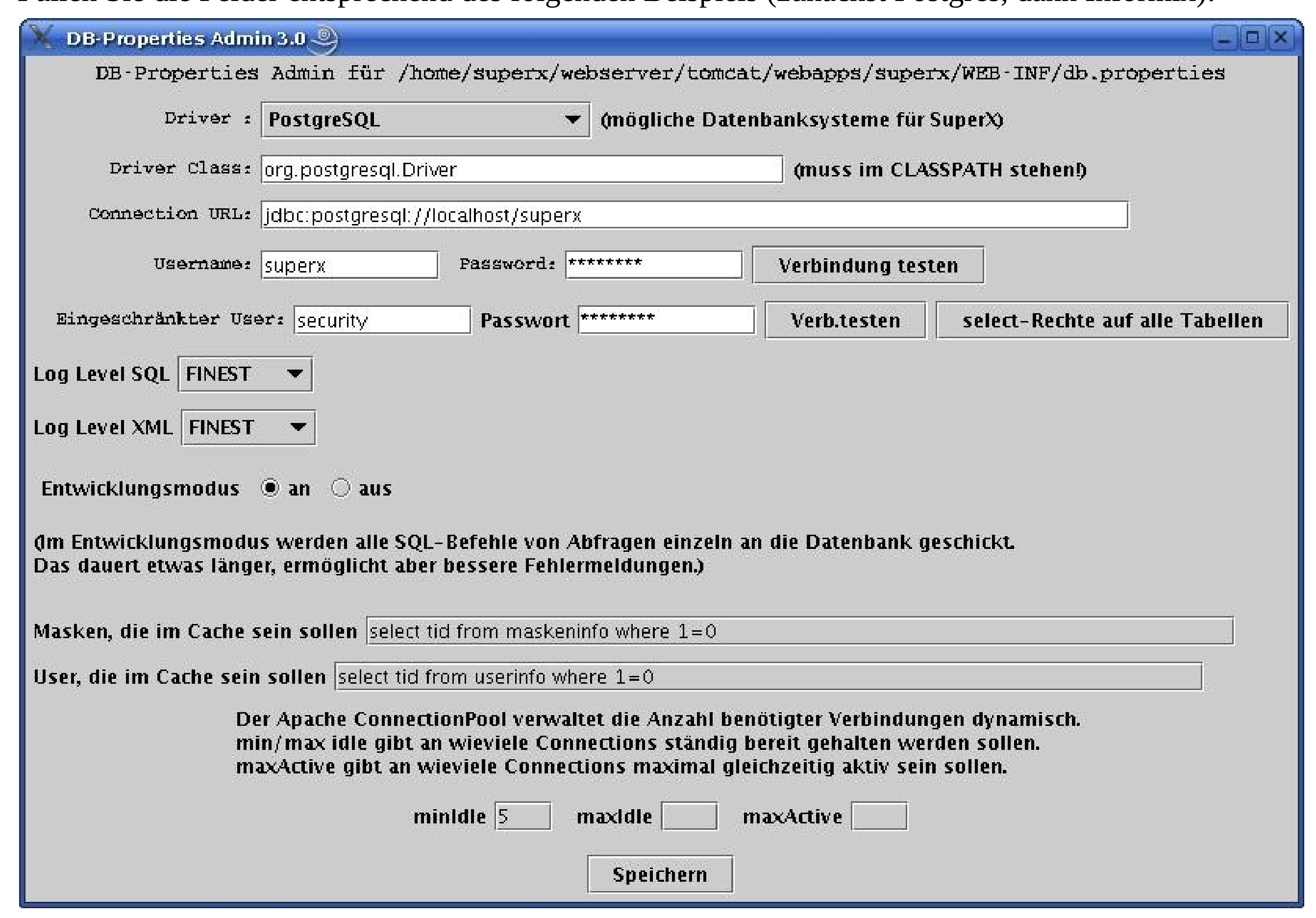
Starten Sie das Tool von einer Konsole bzw. Eingabeaufforderung das Tool mit dem Befehl

`propadmin.bat`

bzw.

`propadmin.x` (unter Linux).

Füllen Sie die Felder entsprechend des folgenden Beispiels (zunächst Postgres, dann Informix):

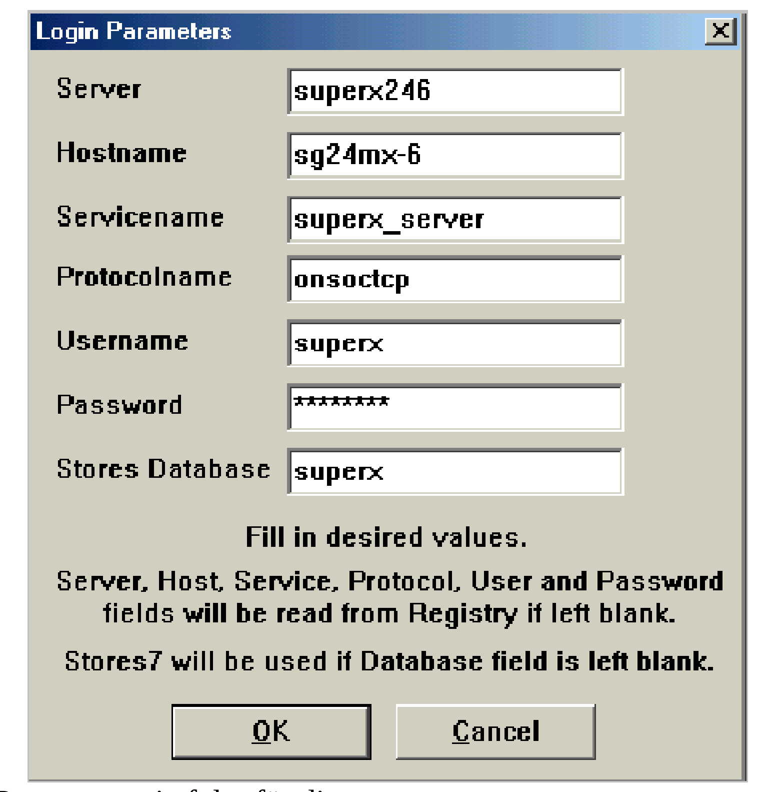


Hinweis für Postgres: Wenn Sie Postgres auf einem anderen Port als dem voreingestellten 5432 betreiben, müssen Sie im jdbc-Treiber als Connection-URL den Port wie folgt angeben:

connectionURL=jdbc:postgresql://localhost:<<Portnumer>>/superx

Bei Informix könnte es beispielsweise so aussehen:

Nehmen wir z.B. für Informix die Parameter beim ILogin (oben erläutert). Wenn die rechte Abbildung eine korrekte Einstellung anzeigt, ....




...dann übernehmen Sie die Parameter wie folgt für die db.properties:

Der Port 1542 ergibt sich aus dem Service für SuperX, der oben bereits beschrieben wurde.

Die Parameter für den **LogLevel** können auf einer Skala von fünf Stufen gewählt werden: FINEST bis SEVERE. Bei FINEST wird fast alles geloggt, bei SEVERE werden nur Fehler geloggt.

Im Entwicklungsmodus werden alle SQL-Befehle von Abfragen einzeln an die Datenbank geschickt.Das dauert etwas länger, ermöglicht aber bessere Fehlermeldungen. Man kann diese Einstellung auch im laufenden Betrieb ändern.

Die Parameter *im Cache* legen fest, wie viel Information gecached werden werden sollen. Standardmäßig wird nichts gecached, aber im Produktiveinsatz sollten hier die entsprechenden Parameter gewählt werden.

In den Connection-Pool Angaben wird angegeben, wieviele Verbindungen maximal gleichzeitig vom Servlet zur Datenbank hergestellt werden sollen.

Durch Anklicken von **OK** wird die Datei `db.properties` (bzw. der Pfad zum Inhalt der Umgebungsvariable DB_PROPERTIES) erstellt, wobei das Passwort verschlüsselt wird. Vorher sollten Sie mit "Verbindung Testen" prüfen, ob eine Datenbankverbindung hergestellt werden kann. Wenn dies nicht klappt, sollten die Fehlermeldungen weiterhelfen.

Wenn Sie einen UNIX / LINUX-Server für Tomcat betreiben wollen, dann ist es möglich, daß Sie unter Linux keine graphische Java-Umgebung starten können. In diesem Fall müssen Sie das Kernmodul auf einem Rechner mit installiertem Java und graphischer Umgebung kopieren, das Programm dort aus der Konsole starten und die Parameter ändern (wichtig: der Rechner muss die gleiche Zeichenkodierung haben, also LATIN1). Danach kopieren sie die Datei `db.properties` mit scp / WinSCP auf den UNIX-Rechner. Alternativ können Sie die Parameter mit dem vi bearbeiten. Wenn der Propadmin ohne graphische Umgebung gestartet wird, kann lediglich das Passwort eingegeben werden.

Wenn Sie Tomcat auf einem anderen Rechner als dem Datenbankserver betreiben, müssen Sie die Startdateien `propadmin.bat` bzw. `propadmin.x` im Verzeichnis

`$SUPERX_DIR/webserver/tomcat/webapps/superx/WEB-INF`

benutzen (in diesem Falle ist das Verzeichnis $SUPERX_DIR/db nicht notwendig.)

#### 2.1.4.4.5Datenbankverbindung und Steuerung von DBForms

Neben der normalen Properties-Konfiguration muss außerdem der Verbindungsparameter für die Servlets von DBFORMS gesetzt werden.

Die zentrale Steuerungsdatei heißt `dbforms-config.xml` und liegt im Verzeichnis `SUPERX_DIR/webserver/tomcat/webapps/superx/WEB-INF`. Dort liegt bereits ein Muster mit dem Namen `kern_dbforms-config_pg.xml` für Postgres bzw. `kern_dbforms-config_ids.xml` für Informix. Diese Datei wird bei der Installation automatisch kopiert nach `dbforms-config.xml`.

Die Datenbankverbindung wird in der server.xml konfiguriert. und am Ende der Datei die Connection-Attribute angeben. Die Parameter sind identisch mit denen, die Sie in der db.properties angeben.

Detaillierte Dokumentation zum Connection Logging sowie allgemein zu DBForms (leider nur in Englisch) finden Sie im beigefügten DBForms-Handbuch im Verzeichnis `doc/dbforms` des Kernmoduls.

Wenn Sie die DBFORMS-Komponente nicht benötigen bzw. aus Sicherheitsgründen für eine externe Website abschalten wollen, gehen Sie wie folgt vor:

| Aktion | Code |
|---|---|
| Sperren Sie das dbforms-Servlet in der Datei `$SUPERX_DIR/webserver/tomcat/webapps/superx/WEB-INF/web.xml`, indem Sie die rechts blau markierten Kommentarzeichen um die entsprechenden Elemente setzen. | ...<br><!--=========== DbForms Controller Servlet ===============--><br>**<!--**<servlet><br><servlet-name>control</servlet-name><br><servlet-class>org.dbforms.servlets.Controller</servlet-class><br><br><init-param><br><param-name>maxUploadSize</param-name><br><param-value>80000</param-value><br></init-param><br></servlet> **-->**<br><br><!--=========== DbForms FileServlet<br>=====================--><br>**<!--**<servlet><br><servlet-name>file</servlet-name><br><servlet-class>org.dbforms.servlets.FileServlet</servlet-class><br><load-on-startup>2</load-on-startup><br></servlet>**-->**<br>...<br><!--==== Controller Servlet and FileServlet Mappings========--><br>**<!--**<servlet-mapping><br><servlet-name>control</servlet-name><br><url-pattern>/servlet/control</url-pattern><br></servlet-mapping><br><br><servlet-mapping><br><servlet-name>file</servlet-name><br><url-pattern>/servlet/file</url-pattern><br></servlet-mapping> **-->** |
| Fügen Sie an das Ende der `web.xml` vor dem End-Tag "</web-app>" folgende Elemente ein | ...<br>**<error-page>**<br>**<error-code>500</error-code>**<br>**<location>/error.htm</location>**<br>**</error-page>**<br></web-app> |
| Ändern Sie am Ende der Datei `$SUPERX_DIR/webserver/tomcat/webapps/superx/WEB-INF/dbforms-config.xml` beim Element **dbconnection** den Attributnamen "name" auf einen nicht existenten Namen, z.B. "superx1". | <dbconnection id="superx" isJndi="true"<br>name="java:/comp/env/jdbc/superx**1**"/> |
| Starten Sie Tomcat neu, und prüfen Sie in der Logdatei `$SUPERX_DIR/webserver/tomcat/logs/catalina.out`, ob der Tomcat- | |

| Start erfolgreich war. | |
|---|---|

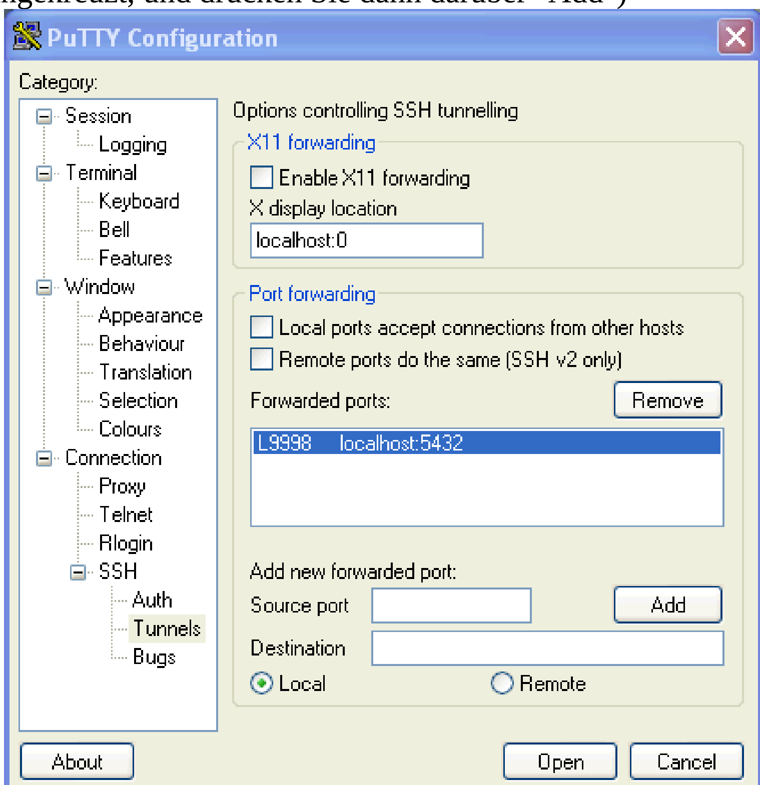
Durch diese Maßnahme sind der DBFORMS-Komponente keine Datenbankverbindungen mehr möglich, und das Ausspähen geschützter Dateien in Tomcat-Systemverzeichnissen durch das Control-Servlet ist nicht mehr möglich.
Eine Abschaltung der DBFORMS beeinträchtigt in keiner Weise die "normalen" Funktionen zur Berichtserstellung von SuperX.

#### 2.1.4.4.6Ein SSH-Tunnel für die Datenbank

Mit der oben beschriebenen Installation ist die Datenbankverbindung zwischen Client und Server noch unverschlüsselt. Zur Verschlüsselung kann einerseits die native Verschlüsselung im DBMS eingeschaltet werden. Man kann aber auch Datenbankverbindungen durch einen zusätzlichen **ssh-Tunnel** verschlüsseln. Zum Tunneln z.B. von Postgres von einem entfernten Rechner über ssh gehen Sie wie folgt vor:

**Unter Windows:**

- Starten Sie den ssh-Client putty (z.B. von http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html)
- Erzeugen Sie ggf. eine neue Session, indem Sie auf der obersten Seite "Sessions" den Host Name, Port, Protocol=ssh eintragen.
- Geben Sie dann bei saved sessions einen neuen Namen, und drücken Sie sicherheitshalber "Save".
- Geben Sie im Menüpunkt "Connection" bei "Auto-Login username" den Namen "superx" an.
- Geben Sie im Menüpunkt "SSH"->"Tunnels" unten im Menü "Add new forwarded Ports"
- bei "Source Port" z.B. "9998" ein.
- Bei "Destination" geben Sie "localhost:5432" ein (wenn 5432 der Port ist, auf dem Postgres läuft).
- Lassen Sie "Local" angekreuzt, und drücken Sie dann darüber "Add")




- Dann speichern Sie die Session auf der obersten Seite "Sessions"
- Dann drücken Sie unten "Open" und loggen sich ein.

**Unter Unix/Linux:**

Geben Sie einfach in der Shell ein:

```
ssh superx@<<IP-Nr. des DB-Servers>> -L 9998:localhost:5432
```

In diesem Moment ist der Tunnel eingerichtet. Sie können ihn nun nutzen, wenn Sie mit Ihrem JDBC- oder ODBC-Client auf den Port localhost:9998 zugreifen.

Z.B. für die sqlWorkbench unter Postgres im Dialog "Connect" die URL

jdbc:postgresql://localhost:9998/superx

Der Tunnel wird geschlossen wenn Sie sich ausloggen. Sie müssen übrigens nicht den Hostnamen des Client-Rechners in die pg_hba.conf eintragen, für Postgres verhält sich der Tunnel so, als ob vom Rechner "localhost" auf den Server zugegriffen wird. Auch in der Firewall des DB-Servers muss nur der SSH-Port freigeschaltet sein, nicht der Datenbank-Port.

Bei Problemen ist ggf. im SSH-Server das Port-Forwarding aus Sicherheitsgründen ausgeschlossen. Für Informix haben wir das obige Vorgehen noch nicht getestet.

### 2.1.5Installation und Pflege des Webservers

Die Servlet-Engine Tomcat verfügt zwar über einen kleinen "eingebauten" Webserver, doch für den Echtbetrieb sollte man aus Performance-Gründen einen der marktgängigen Webserver nutzen (z.B. Apache, IIS), der auch Verschlüsselung bietet. Für den Echtbetrieb empfehlen wird die Installation eines Apache 1.3.x auf Linux-Basis – meist ist dieser in der Linux-Distribution bereits integriert. Der Apache läßt sich sehr gut mit dem Tomcat verbinden (siehe Tomcat User's Guide im Kernmodul unter $SUPERX_DIR/doc/tomcat/doc). Bei der Linux-Installation gehen wir davon aus, dass alle Maßnahmen unter der Kennung superx erfolgen, und dass der User superx Zugriffsrechte auf die Datenbank hat. Beim Kopieren des Archivs sollten Sie darauf achten, dass der User superx auf die Scriptdateien Ausführungsrechte besitzt.

Die folgenden Anleitungen gehen davon aus, dass Sie als Installationspfade für den Webserver C:\superx\webserver (unter win32) und /home/superx/webserver (unter UNIX / LINUX) gewählt haben. Sie können natürlich auch andere Pfade wählen, müssen dann aber die Pfade in dieser Dokumentation entsprechend umsetzen. Fehlende oder falsche Pfade bzw. Umgebungsvariablen sind in Java- und Datenbankprojekten eine wichtige Fehlerquelle (z.B. unter LINUX die Groß- / Kleinschreibung). Aus diesem Grunde haben wir ins Stammverzeichnis des webservers eine html-Datei erstellt ( $SUPERX_DIR/webserver/index.htm), von der aus die Parameter und Pfade schrittweise überprüft werden können.

#### 2.1.5.1Installation von Java und Datenbanktreibern

Der Webserver muss Java-fähig sein, damit er Servlets ausführen kann.

Arbeitsschritte:

1.Java Development Kit JDK StandardEdition 1.6.x (oder höher) installieren

- am besten alles ins Verzeichnis c:\Programme\jdk1.6.x bzw. /usr/local/jdk1.6.x

2.Die Umgebungsvariable JAVA_HOME setzen und das bin-Verzeichnis der Java-Installation in den PATH legen. Die Umgebungsvariable CLASSPATH sollte mindestens "." enthalten, aber auf keinen Fall einen älteren XML-Parser (z.B. xerces 1.0).
3.Nur für Informix-Anwender: Laden Sie den jdbc-Treiber von Informix (oder das Informix Client SDK) herunter, installieren Sie das Produkt und kopieren Sie die Datei ifxjdbc.jar nach `$SUPERX_DIR/webserver/tomcat/webapps/superx/WEB-INF/lib`.
4.Testen Sie die Datenbankverbindung mit dem Werkzeug propadmin.
5.Ablauf mit einem einfachen java-Servlet testen

#### 2.1.5.2Einrichtung der Servlet-Engine

Die Servlet-Engine ermöglicht dem Webserver das SuperX-Servlet auszuführen. Anders als andere Scriptsprachen (z.B. asp, PHP, Perl) für Webserver ist der Java-Code als Bytecode kompiliert; die Servlets werden normalerweise also nicht auf dem Webserver entwickelt und getestet, sondern auf einem eigenen Entwicklungsrechner.

Es gibt verschiedene Anbieter von Servlet-Engines, teilweise kostenpflichtig, z.B.

- Unify ServletExec
- Allaires Jrun
- Apaches JServ
- Apache / Jakartas Tomcat

Im Kernmodul ist der Tomcat 7.0.20 mitgeliefert. Tomcat ist von SUN als Referenzimplementierung von Webapplikationen anerkannt, d.h. Sie sollten die Konfiguration mühelos auf andere Server übertragen können. Die Web-Applikation von SuperX läuft unter allen Tomcat-Versionen.

Tomcat ist eine kostenlose und gleichzeitig umfassende Engine, die darüber hinaus auch recht leicht zu installieren ist und auf vielen gängigen Webservern läuft (Apache[9], IIs, Netscape). Sie ist im Rahmen des Apache-Projektes frei verfügbar und distribuierbar, deshalb ist Tomcat in der SuperX-Distribution bereits enthalten. Wenn Sie das Kernmodul entpacken, ist Tomcat mitsamt dem SuperX-Kontext bereits installiert. Sie müssen nur noch ein paar Schritte durchführen.

##### 2.1.5.2.1Steuerung des Servers: Die server.xml

Editieren Sie zunächst die Konfigurationsdatei `$SUPERX_DIR/webserver/tomcat/conf/server.xml`.

Hier werden die Ports und Anbindungen der Tomcat-Implementation angepasst. Standardmäßig läuft Tomcat auf dem Port 8080, und die Apache-Anbindung auf dem Port 8009. Weiterhin muss der Port 8005 für den Shutdown frei sein. Die Apache-Anbindung ist weiter unten dokumentiert.

Wichtig beim Betrieb des Tomcat mit UTF-8-Codierung: Der jew. Connector muss das weitere Attribut `URIEncoding="UTF-8"` aufführen. Wenn z.B. der Connector 8080 genutzt wird, sieht das so aus:

[9] Eine hervorragende Einführung zum Einsatz des Apache mit Tomcat findet sich in der Zeitschrift iX, 2/2001, S.48ff.

**Einrichtung des Connectors**

```
<!-- Define a non-SSL HTTP/1.1 Connector on port 8080 -->
<Connector port="8080" maxHttpHeaderSize="8192"
maxThreads="150" minSpareThreads="25" maxSpareThreads="75"
enableLookups="false" redirectPort="8483" acceptCount="100"
connectionTimeout="20000" disableUploadTimeout="true"
URIEncoding="UTF-8"/>
```

Dies ist wichtig für den Ajax-Client.

### 2.1.5.2.2Datenbankverbindung für DBFORMS: die context.xml

Die Datenbank-Verbindung für DBFORMS wird in der Datei `$SUPERX_DIR/webserver/tomcat/webapps/superx/META-INF/context.xml` definiert. Diese Datei wird im Kernmodul nicht ausgeliefert, es existiert aber einer Musterdatei `context.xml.sam` im gleichen Verzeichnis. Die Datei sähe dann z.B. für Postgres auf dem Rechner "miles" auf Port  5432 so aus:

**Der SuperX-Kontext in der context.xml**

```
<Context docBase="${catalina.home}/webapps/superx" debug="0"
                 reloadable="true" crossContext="true">
          <Logger className="org.apache.catalina.logger.FileLogger"
                     prefix="localhost_superx_log." suffix=".txt"
              timestamp="true"/>

          <!--
          <Manager className="org.apache.catalina.session.PersistentMana-
ger"
              debug="0"
              saveOnRestart="true"
              maxActiveSessions="-1"
              minIdleSwap="-1"
              maxIdleSwap="-1"
              maxIdleBackup="-1">
                <Store
className="org.apache.catalina.session.FileStore"/>
          </Manager>
          -->
          <Environment name="maxExemptions" type="java.lang.Integer"
                      value="15"/>
          <Parameter name="context.param.name" value="context.param.va-
lue"
                     override="false"/>
          <Resource name="jdbc/superx" auth="Container"
          type="javax.sql.DataSource"
        factory="org.apache.commons.dbcp.BasicDataSourceFactory"
        driverClassName="org.postgresql.Driver"
        url="jdbc:postgresql://localhost/superx"

      username="superx"
      password="anfang12"
      maxActive="7"
      maxIdle="5"
      maxWait="1"
      removeAbandoned="true"
      removeAbandonedTimeout="10"
      />
```

Die von Ihnen anzupassenden Werte sind jeweils **fett** gedruckt. Leider verlangt **dbforms**, dass das Passwort im Klartext eingegeben wird; achten Sie daher darauf, dass die Datei nur für den User selbst lesbar

ist. Alternativ können Sie (zumindest bei Postgres) über die hba_conf steuern, wie der User sich ohne Passwort authentifizieren kann.

In der Musterdatei befindet sich auch ein Beispiel für Informix.

#### 2.1.5.2.3Die Datei conf/web.xml

In der Datei conf/web.xml definieren sie u.a. die serverweite "Welcome-Page" bzw. deren Reihenfolge., welche wiederum Dateien anzeigen, wenn der Anwender ein Verzeichnis ohne Dateinamen aufruft, z.B. http://servername/superx/:

**Welcome-Files für Tomcat-Verzeichnisse**

```
<welcome-file-list>
        <welcome-file>index.html</welcome-fi-
le>
        <welcome-file>index.htm</welcome-file>
        <welcome-file>index.jsp</welcome-file>
    </welcome-file-list>
```

Wenn Sie z.B. die Reihenfolge so ändern, dass zuerst die Datei index.jsp angezeigt wird (sofern sie existiert), dann können Sie eigene "Homepages" definieren, die nicht von der SuperX-Distribution (z.B. bei Updates) überschrieben würden. Außerdem ist dies eine sinnvolle Sicherheitsmassnahme, weil so keine Directory Listings angezeigt werden.

Änderungen in der Datei web.xml in der Webanwendung ( <<Webanwendung>>/WEB-INF/web.xml) überschreiben die Einträge in der serverweiten web.xml.

Weitere Konfigurationsmöglichkeiten (Server Side Includes etc). sind in dieser Datei dokumentiert.

Vergleiche auch den unten folgenden Abschnitt zur Einrichtung der SuperX-Servlets unter Tomcat.

#### 2.1.5.2.4Administrator und Manager

Die Voreinstellungen in Tomcat 4 sind für einen ersten Testbetrieb bereits vorbereitet. Bearbeiten Sie lediglich die Datei conf\tomcat-users.xml

Im folgenden Beispiel wird der User superx mit dem Passwort "anfang12" als Admin und als Manager eingetragen.

**Tomcat Users: Administrator und Manager**

```
<?xml version='1.0' encoding='utf-8'?>
<tomcat-users>
  <role rolename="tomcat"/>
  <role rolename="role1"/>
  <role rolename="manager"/>
  <role rolename="admin"/>
  <user username="tomcat" password="tomcat" roles="tomcat"/>
  <user username="role1" password="tomcat" roles="role1"/>
  <user username="superx" password="anfang12"
roles="tomcat,admin,manager"/>
</tomcat-users>
```

Natürlich ist dieses nur ein Beispiel für eine erste Testimplementation, nicht für einen produktiven Server geeignet.

#### 2.1.5.2.5 Einrichten der SuperX-Servlets unter Tomcat

Anpassen der Datei **db.properties** mit den Datenbank-Zugangsdaten (siehe **Anpassen der Datenbankparameter für das SuperX-Servlet**) ist Voraussetzungen dafür, dass der Webserver auf die Datenbank zugreifen kann.

Schließlich muss man in der Datei
$SUPERX_DIR/webserver/tomcat/webapps/superx/web-inf/web.xml
Einträge für die SuperX-Servlets anpassen, wenn man die Standardvorgaben nicht übernehmen will.

Für das Applet:

Wenn das Applet verwendet werden soll, ist ein Eintrag für das SuperXDBServlet nötig.

Der Parameter max_rows ganz amEnde legt fest, wie viele Datensätze ein Servlet maximal an den Client ausliefert. Wenn Ihr Organigramm z.B. mehr als 3000 Sätze enthält, dann sollten Sie diesen Wert höher setzen.

Auszug aus der Web.xml

```
<servlet>
      <servlet-name>SuperXDBServlet</servlet-name>
      <servlet-class>SuperXDBServlet </servlet-class>

 <init-param>
      <param-name>max_rows</param-name>
             <param-value>3000</param-value>
</init-param>   </servlet>
```

Wenn das Applet nicht eingesetzt wird, können Sie es deaktivieren (vergl. Checkliste Sicherheitsmaßnahmen, Applet deaktivieren).

Zentrales Servlet ist der SuperXManager, für den ein Eintrag benötigt wird.

```
<servlet>
        <servlet-name>
            SuperXManager
        </servlet-name>
        <servlet-class>
            de.superx.servlet.SuperXManager
        </servlet-class>
 <!–-Bei Bedarf kann Saxon-als XSL-Prozessor definiert werden, wenn der
folgende Eintrag aktiviert wird-->
<!--<init-param>
<param-name>xsl_processor</param-name>
<param-value>net.sf.saxon.TransformerFactoryImpl</param-value>
</init-param>   -->
<!–Die maximale Anzahl von Datensätzen, die eine Abfrage zurückliefern
sollte, sollte jetzt beim SuperX-Manager angegeben werden, nicht mehr beim
SuperXDBServlet, im Normalfall reicht der Standardwert von 20000 der ohne
init-param als default genommen wird
<init-param>
<param-name>maxRows</param-name>
<param-value>20000</param-value>
</init-param>
<!–-neu in 3.5rc2 – Erläuterung s.u.-->
<init-param>
<param-name>field1Cache</param-name>
<param-value>tid&gt;10000</param-value>
</init-param>
```

```
    <load-on-startup>50</load-on-startup>
  </servlet>
```

**field1Cache**

Neu in SuperX3.5rc2 ist die Möglichkeit einen sogenannten field1Cache für Auswahllisten (Feldart 1) zu nutzen.

Wenn ein entsprechender init-param beim SuperXManager definiert ist, lädt sich der webserver beim Start Inhalte für die angegeben Felder der Feldart 1 (aus felderinfo), in denen es keine dynamischen Tags gibt (wie z.B. <<Haushaltsjahr>>) in einen Cache. Dadurch wird der Start des Webservers natürlich etwas langsamer, aber wenn die Benutzer einzelne Maske aufrufen, können diese schneller dargestellt werden, weil weniger Datenbankzugriffe nötig sind.

Als Param-value muss eine where-Bedingung für einen select auf die Tabelle felderinfo angegeben werden. Sie können das Beispiel tid>10000 belassen oder bei Bedarf bestimmte Felder auslassen, z.B.

```
tid&gt;10000 and name not in ('Haushaltsjahr','Semester').
```

Der Cache wird aktualisiert, wenn im SuperXManager-Servlet auf den Button “Server-Cache aktualisieren” geklickt wird oder der Webserver neu gestartet wird. Außerdem wird er jeden Morgen einmal automatisch aktualisiert. Felder die sich zusätzlich zu den nächtlichen Updates dynamisch ändern, sollten ausgeschlossen werden, damit sie immer aktuell aus der Datenbank geholt werden.

Ein weiterer Parameter für die gesamte Webapplikation, der aber nur im XML-Frontend ausgewertet wird, lautet <session-timeout> (siehe Beispiel-web.xml in unserem Kernmodul, ganz am Ende der webapp-Deklaration). Dieser Wert beschreibt die "Lebenszeit" einer Anmeldung bei Inaktivität des Benutzers (in Minuten). Ein negativer Wert bedeutet, dass die Session nie beendet wird. Ein sinnvoller Wert ist z.B. 180 (3 Stunden). Je länger die Zeit, desto höher die Belastung des Servers.

Sie können auch durch spezielle Fehlerseiten die normale Fehlerausgabe des SuperX-Servlets sperren.

**Fügen Sie an das Ende der `web.xml` vor dem End-Tag "</web-app>" z.B. folgende Elemente ein**

```
...
<error-page>
<error-code>500</error-code>
<location>/error.htm</location>
</error-page>
</web-app>
```

Die ist die Voreinstellung bei Neuinstallation von SuperX, ältere Installationen müssen dies ggf. nachholen.

Sie können auf verschiedene Fehler-Codes sowie Exception-Types eigene Fehlerseiten definieren. Details dazu finden Sie in der Dokumentation Ihres Applikationsservers.

### 2.1.5.2.6 Start des Tomcat

Vor dem Start müssen die Umgebungsvariablen der Datei `$SUPERX_DIR/db/bin/SQL_ENV` geladen werden.

**Die Umgebungsvariable JAVA_HOME muss korrekt gesetzt sein**

- **Unter WIN32:**
  Das geht unter MS-DOS als Kommandozeile `set JAVA_HOME=c:\jdk1.6x` oder man macht einen Eintrag als Systemvariable (Systemsteuerung – System – Erweitert – Umgebungsvariablen) neue Systemvariable JAVA_HOME , Wert c:\jdk1.6x
  (wenn nur die Runtime installiert ist, ist das Verzeichnis evtl. c:\programme\javasoft\jre\1.6x)
- <tomcat-Basisverzeichnis>\bin\startup.bat ausführen (zum Beenden shutdown.bat)
- Falls unter Windows 98/ME eine Meldung kommt, dass der Umgebungsspeicher nicht ausreicht, muss man über start->Ausführen folgende Zeile eingeben: `command.com /p /e:4096`
- **Unter UNIX / LINUX:**
  Setzen Sie entweder in der /etc/profile oder in der Datei `.profile` bzw. .bashrc im Heimverzeichnis des Users `superx` bzw. bei Betrieb von Datebank und Webserver auf einem Rechner in der Datei `$SUPERX_DIR/db/bin/SQL_ENV` die Zeile ein:
  `export JAVA_HOME=/usr/lib/java` (als Beispiel für eine Java-Installation unter SUSE Linux 9.1)
- Das aktuelle Verzeichnis sollte im PATH sein (ggfs. /etc/profile oder .profile bzw .bashrc
  `PATH=PATH$:.;export PATH`
- Melden Sie sich ab und wieder an
- `<tomcat-Basisverzeichnis>\bin\startup.sh` ausführen (zum Beenden `shutdown.sh`).

Das Terminal-Fenster zeigt den Port an, auf dem Tomcat läuft, z.B. 8080; um die Engine zu testen, kann man einen Webbrowser (zur Not auch ) starten und die Seite ... aufrufen.

- Testen, ob der SuperX-Kontext unter Tomcat verfügbar ist:
  http://localhost:8080/superx/
- Testen, ob Sie sich auf der SuperX-Datenbank anmelden können
  http://localhost:8080/superx/xml/
- Testen, ob das Applet läuft
  http://localhost:8080/superx/applet/

Beendet wird Tomcat mit dem Befehl: `shutdown.bat` für MS-DOS bzw. `shutdown.sh` für UNIX.

•

### 2.1.5.2.7Die Übertragung der Web Application

Wenn Sie die SuperX-Webapplikation auf einem vorhandenen Tomcat installieren wollen, müssen Sie alle Libraries (*.jar) von der SuperX-Distribution kopieren und ältere Versionen, die bereits vorhanden sind, löschen (Wichtig!). Außerdem dürfen Sie auf dem Datenbankserver nicht "unseren" Tomcat löschen, selbst wenn er nicht gebraucht wird: Die Java-Bibliotheken und die properties-Dateien werden auch von Scripten auf dem Datenbankserver benötigt.

Sie kopieren nun das gesamte Verzeichnis `$SUPERX_DIR/webserver/tomcat/webapps/superx` in das `webapps`-Verzeichnis des Tomcat. Wenn  Sie Tomcat 5.5 oder höher nutzen, ist eine Übertragung der Webanwendung problem möglich. Wenn nicht, dann müssen Sie z.B. nach `web_tomcat3.xml` sichern, und die Datei `web_tomcat4.xml.sam` nach `web.xml` kopieren. Gegebenenfalls müssen Sie dann Steuerungsparameter in der Web-Application in der Datei `WEB-INF/web.xml` prüfen (z.B. maxRows oder das sql- bzw. connection-Logging, Session-Timeout).

#### *2.1.5.2.7.1Übertragung der Webapplikation auf einen vorhandenen Tomcat unter Windows*

Der Betrieb von Tomcat 4.x-6.x unter Windows ist grundsätzlich möglich. Wenn Tomcat unter **cygwin** installiert wird, entsprechen alle Schritte dem obigen Vorgehen unter Linux. Wenn Tomcat aus dem exe-Installer als Dienst installiert wird, dann müssen zwei Unterschiede beachtet werden:

- Die Standardausgabe von Tomcat geht nicht nach logs\catalina.out, sondern stdout_<<Datum>>.log bzw. stderr_<<Datum>>.log.
- Der Pfad zur Logging-Datei für DBFORMS wird in der Datei WEB-INF/log4j.properties festgelegt. Hier wird die Pfadangabe nicht relativ zum Statup-Verzeichnis von Tomcat gegeben, sondern absolut, z.B. log4j.appender.logFile.File=C:/tmp/dbforms.log

Bei manchen Systemen mit Java 1.6.x startet Tomcat 5.5 nicht als Dienst, und in der Datei tomcat\logs\jakarta-service*.log steht etwas wie:

```
[174  javajni.c] [error] Das angegebene Modul wurde nicht gefunden.
[947  prunsrv.c] [error] Failed creating java C:\Programme\Java\jre1.6.0_01\bin\client\jvm.dll
```

In diesem Falle muss die Datei %JAVA_HOME%\bin\msvcr71.dll in das Verzeichnis c:\windows\system32 kopiert werden.

#### *2.1.5.2.7.2Übertragung der Webapplikation auf einen vorhandenen Tomcat 5.5*

Die Übertragung der Webanwendung auf die Referenzimplementation Tomcat 5.5 ist problemlos möglich. Kopieren Sie die Webapplikation superx in Ihren Tomcat ins Verzeichnis tomcat/webapps, und starten Sie Tomcat neu. Wir empfehlen bei Problemen, zunächst den mit SuperX ausgelieferten Tomcat zu nutzen, und erst wenn SuperX hier problemlos läuft, die Webanwendung auf einen anderen Tomcat zu übertragen.

### 2.1.5.2.8Das SuperXManager-Servlet

Mit dem SuperXManager-Servlet kann man verschiedene Einstellungen vornehmen.
Es kann von Admins aufgerufen werden unter der Adresse:
http://rechnername:port/superx/servlet/SuperXManager

**Server-Cache**

SuperX cacht zur Performanceverbesserungen einige Dinge im Webserver, dazu gehören Erläuterungen und Übersetzungen und für's XML-Frontend auch: User,Userrechte und Sichten und auch Abfragen wenn in der db.properties eingetragen.

Falls Sie bei Entwicklungsarbeiten Änderungen an diesen Dingen gemacht haben und im XML-Frontend arbeiten, müssen Sie einmal den "Server-Cache aktualisieren". Hinweis: Neue Sichten können z.B. auch durch ein Update der COB-Daten erfolgen, wenn neue alternative Hierarchien dazukommen.

**Entwicklungsmodus**

Im Entwicklungsmodus werden alle SQL-Befehle von Abfragen einzeln an die Datenbank geschickt. Das dauert länger, ermöglicht aber bessere Fehlermeldungen. Die Standardeinstellung ist in den db.properties hinterlegt. Sie kann hier bei Entwicklungsarbeiten umgestellt werden.

**Achtung:** Unter Informix funktionieren einige Maske nicht, wenn der Entwicklungsmodus ausgeschaltet ist. Zur Sicherheit sollten sie ihn hier eingeschaltet lassen.

Mit **Logdateien leeren** können Sie die superx_*.log-Dateien im Tomcat/logs-Verzeichnis leeren. Dies kann bei Entwicklungsarbeiten praktisch sein, wenn Sie nur die Logs eines bestimmten Vorgangs haben möchten.

Außerdem werden noch verschiedene SQL/XML-Loginformationen für Entwicklungszwecke angezeigt.

#### 2.1.5.2.9Verbesserung der Performance

Die Tomcat-Performance läßt sich durch Zuweisung von mehr RAM steigern. Dazu muss lediglich die Umgebungsvariable `JAVA_OPTS` gesetzt werden, z.B. mit

```
JAVA_OPTS= "-Xmx300M -Djava.awt.headless=true"
export JAVA_OPTS
```

Hierdurch werden 300 MB RAM dem Tomcat zugewiesen. Die Umgebungsvariable wird außerdem auch von diversen SuperX-jdbc-Clients berücksichtigt. Dies ist z.B. sinnvoll, wenn größere Tabellen be- oder entladen werden. Der Passus `" -Djava.awt.headless=true"` muss immer dabei sein, wenn Tomcat auf einem UNIX-System ohne graphische Konsole aus aufgerufen wird.

Die Performance von Tomcat läßt sich weiterhin durch den Lastausgleich in Kombination mit dem Apache Webserver steigern. Beim Tomcat 3.2.x Die Konfiguration wird in der Datei `conf/workers.properties` vorgenommen.

Die bereits vorhandenen Beispieleinträge sollten die Konfiguration des Lastausgleich erläutern. Weitere Details zur `workers.properties` finden Sie in der Anleitung zur Anbindung an den Apache. Die verschiedenen Howtos in der Tomcat-Distribution erläutern Details zur Apache-Anbindung.

#### 2.1.5.2.10Einrichtung einer SSL-Verbindung in Tomcat

Wenn Tomcat mit einer SSL-Verschlüsselung arbeiten soll, dann sollte von vornherein das JDK 1.4.x installiert werden, weil dies die notwendigen Bibliotheken dazu enthält. Eine bestehende JDK-1.3.Installation kann aber genutzt werden, wenn die Verschlüsselungs-Klassen von Java (jsse.jar, jnet.jar,jcert.jar) nach kopiert werden. Es gibt ferner die Möglichkeit, die Verschlüsselung vom Webserver (Apache oder IIS) vornehmen zu lassen (siehe Tomcat-Dokumentation in tomcat-docs/tomcat-ssl-howto.htm); da SuperX aber kaum statische html-Seiten enthält, empfehlen wir immer,die Verschlüsselung direkt in Tomcat einzurichten. Dies ist relativ einfach, wenn kein öffentlich bekanntes und signiertes Zertifikat genutzt werden soll.

Erzeugen Sie auf dem Webserver einen Keystore mit dem Befehl

| | |
|---|---|
| **Windows:** | `%JAVA_HOME%\bin\keytool -genkey -alias tomcat -keyalg RSA` |
| **UNIX:** | `$JAVA_HOME/bin/keytool -genkey -alias tomcat -keyalg RSA` |

Die Parameter werden erfragt; wichtig ist, dass der erste Eintrag (Vor- und Nachname, COMMON NAME CN) der DNS-Name des Werbservers ist, z.B. `superx.verwaltung.uni-duisburg.de`. Als Passwort geben Sie beide Male "changeit" an. Draufhin wird ein Zertifikat erzeugt und in der Datei `.keystore` im Homeverzeichnis des Benutzers angelegt (unter Windows im Profiles-Verzeichnis, unter UNIX im home-Verzeichnis).

Das persönliche Zertifikat können Sie durch einen kommerziellen Zertifizierungsserver publizieren; zu Testzwecken können Sie auch ein selbsterstelltes Zertifikat erzeugen:

```
keytool -selfcert -alias tomcat -validity <<Anzahl der Tage>>
```

Danach ändern Sie die Datei `$TOMCAT_HOME/conf/server.xml`, indem Sie die Passage mit der SSL-Verschlüsselung ent-kommentieren und den normalen Port (8080) auskommentieren. Danach ist das Servlet über https://localhost:8443 statt http://localhost:8080 erreichbar. Sie müssen alle Links entsprechend ändern und in der Datei `superx.properties` die Zeile

| | |
|---|---|
| **superx.properties mit ssl** | `SxServerURL=https://localhost:8443/superx/servlet/SuperXDBServlet`<br>statt<br>`SxServerURL=http://localhost:8080/superx/servlet/SuperXDBServlet`<br>aktivieren. |

Das Zertifikat können Sie löschen, indem Sie auf der Kommandozeile eingeben:

```
keytool -delete -alias tomcat
```

#### *2.1.5.2.10.1Signierung eines Zertifikats in Tomcat*

Bei selbst signierten Zeritfikaten erscheint im Browser immer eine Sicherheitswarnung. Um dies zu vermeiden, muss man ein öffentliches Zertifikat von einem Trust Center erwerben. Dies kann man im Apache eintragn (s.u.), aber auch direkt im Tomcat, wenn Sie keinen Apache nutzen[11]:

1.public key + private key erzeugen und die im keystore-file ablegen
(der private key wird dabei mit passwd verschlüsselt):
`keytool -genkey -keyalg RSA -alias tomcat -keystore xxx.jks`

2.certificate request generieren --> Datei server.csr und an die CA schicken:
`keytool -certreq -keyalg RSA -file server.csr -keystore xxx.jks`

3.Den von der CA signierten public key = Serverzertifikat
zurückbekommen --> Datei `server.cer`

4.Zuerst das Zertifikat der CA (z.B. UTN-USERFirst-Network Applications, http://www.usertrust.com) downloaden und in den keystore einspielen:
`keytool -import -file UTN.cer -alias tomcat -keystore xxx.jks`

5.Dann das neue Serverzertifikat in den keystore einspielen:
`keytool -import -file server.cer -alias tomcat -keystore xxx.jks`

6.in der Tomcat-server.xml auf die keystore-Datei verweisen:
`keystoreFile="<<Pfad zur xxx.jks-Datei>>" keystorePass="passwd"`

---

[11] Vielen Dank für diese Anleitung an Herrn Behnke, Uni Bonn. Siehe auch http://www.junlu.com/msg/48529.html

Wichtig: dasselbe passwd einsetzen wie unter 1. zum Verschlüsseln des private key benutzt wurde

7.Restart Tomcat

8.https-Verbindung zum Server, Zertifikat überprüfen - vertrauenswürdig?

#### 2.1.5.2.11Zusätzliche Verschlüsselung im Applet durch Public-Private-Key-Kontrolle

Zur Erhöhung der Sicherheit im **SuperX-Applet** ist es möglich, eine DSA-public/private-Key-Kontrolle zu installieren. Dabei wird jeder SQL-Befehl, der vom Applet ans Servlet geschickt wird, mit dem einen Key signiert und im Servlet wird mit Hilfe des anderen, nur dort bekannten Keys kontrolliert, ob der ankommende SQL eine gültige Signatur aufweist.

Zur Installation eines zufällig erzeugten Key-Paars brauchen Sie auf dem Datenbankserver in der Shell nur die `SQL_ENV` aufzurufen und anschließend das Kommando `sx_keymanager.x install` abzuschicken. Mit `sx_keymanager.x delete` könnten Sie ggfs. das Schlüsselpaar wieder entfernen und mit `sx_keymanager.x check` prüfen, ob ein Schlüsselpaar installiert ist. Wenn Sie Tomcat auf einem separaten Rechner betreiben, brauchen Sie hier kein Script ausführen, es recht, dort das jeweilige Kernmodul-Paket zu entpacken. Bei mandantenfähigen Installationen müssen Sie das Script `sx_keymanager.x install` für jeden Mandanten einzeln ausführen.

Wenn Sie Tomcat neu starten, können Sie in den Logdateien (normalerweise `$SUPERX_DIR/webserver/tomcat/logs/catalina.out`) kontrollieren, ob die public/private key Kontrolle aktiv ist oder nicht.

Nach der Meldung zum Aufbau des Datenbank-Connectionpools kommt ein einsprechender Hinweis.

```
Aufbau des ConnectionPool (....)  .. OK
             public/private key aktiv
```

Im SuperX-Applet können Sie den Info-Button anklicken, in der erscheinenden Infobox wird angegeben, ob public/private key Kontrolle aktiv ist oder nicht.

#### 2.1.5.2.12Tomcat als Dienst unter Linux

Die Implementation von Tomcat als Dienst ist unverzichtbar, damit der Serve rbeim Hochfahren automatisch startet. Wir haben Konfigurationsscripte und Startscripte mitgeliefert, die Sie recht leicht anpassen können.

Im Verzeichnis `$SUPERX_DIR/webserver/etc` befinden sich Musterdateien, um einen Dienst unter SUSE oder RedHat Linux daraus zu machen. Kopieren Sie die Inhalte des Verzeichnisses `etc` als root auf den Webserver ins Verzeichnis `/etc`, und passen Sie `/etc/sysconfig/superx_webserver` entsprechend Ihrer Umgebung an. Schließlich muss ein symbolischer Link von `/etc/init.d/superx_webserver` nach (usr)/bin/rcsuperx_webserver gelegt werden.:

```
ln -symbolic /etc/init.d/superx_webserver  /bin/rcsuperx_webserver
```

Danach kann man den Dienst im Runlevel-Editor des YAST aktivieren (Runlevel 3 und 5). Der Dienst muss vor dem Webserver, aber nach dem Start des Datenbankservers gestartet werden. Der Dienst selbst wird vom User superx gestartet, und kann jederzeit mit

```
rcsuperx_webserver restart
```

neu gestartet werden.

Unter RedHat Linux gibt es ebenfalls Werkzeuge für die Einrichtung der Runlevel, ggf. kann man auch manuell symbolische Links einrichten, wie beim Start des Datenbankservers beschrieben. Außerdem muss ggf. die Umgebung vor dem Start des Tomcat geladen werden, z.B. durch Aufruf der SQL_ENV. Wichtig ist, dass beim Start des Tomcat als Dienst die Variable JAVA_HOME korrekt gesetzt ist und die Variable LANG auf eine deutsche Locale zeigt. Letzteres ist bei RedHat nicht standardmäßig vorgesehen.

Die Einrichtung des Tomcat als Dienst ist auch für Windows-Server möglich, wie im folgenden gezeigt wird.

#### 2.1.5.2.13Tomcat als Dienst unter Windows einrichten (nur WINNT/2000 und Tomcat 3.x)

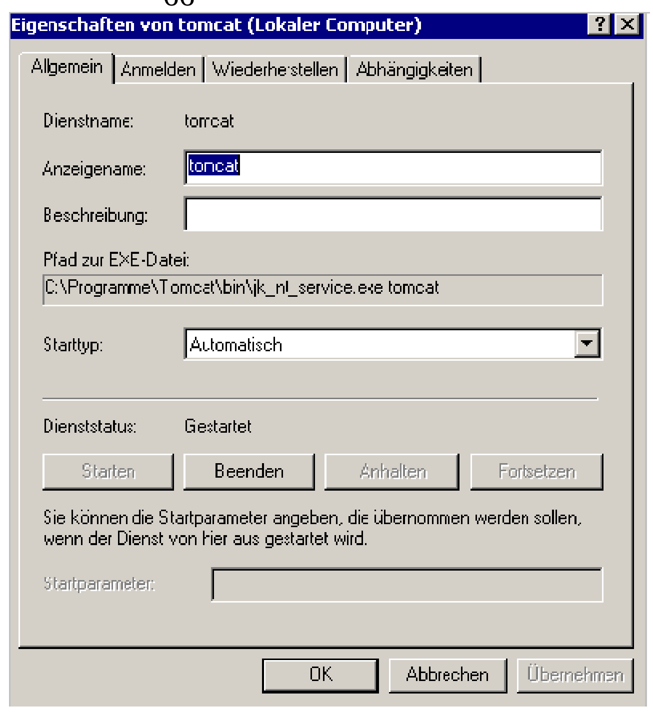
Tomcat muss auf Windown-NT/2000-Rechnern nicht in einer DOS-Box laufen, sondern kann auch als Dienst laufen. Die Installer von Tomcat 7 sehen unter Windows NT /2000/XP eine Installation als Dienst vor.

Unter NT 4 läuft der Tomcat-Dienst nur mit dem JDK 1.6.x, unter Win2000 sollte man Java JDK 1.6.x oder höher installieren. Die Variable JAVA_HOME zeigt dann auf dieses Verzeichnis. Für die Einrichtung muss man bei Windows folgendes machen:

- In der  Datei `d:\superx\webserver\tomcat\conf\wrapper.properties` öffnen und `Tomcat_home` und `Java_Home` auf den richtigen Pfad setzen
- Man fügt der Computerverwaltung Tomcat als Dienst hinzu, indem man in einer DosBox vom `<tomcat>/bin`-Verzeichnis aus "`jk_nt_service –I tomcat ..\conf\wrapper.properties`" ausführt.
- Dann kann man den Dienst über die Systemsteuerung -> Dienste starten (besser: auf automatisch set-zen), und theoretisch läuft Tomcat auch dann, wenn kein User auf dem Rechner angemeldet ist. Aus der DOS-Box kann man den Dienst auch mit `net start tomcat` starten.
- Die Deinstallation des Dienstes erfolgt über `jk_nt_service –R tomcat`

Der Dienst wird in der Systemsteuerung des Rechners aufgeführt, und das Ergebnis sieht unter Win2000 wie folgt aus:

Rechts sehen Sie die Eigenschaften des tomcat-Dienstes unter NT-Server. Bei dem Starttyp können Sie automatisch wählen, und die weiteren Registerkarten sind nicht gefüllt. Der Dienst lässt sich mit den Start/ Unterbrechungsbuttons manuell neu starten:




### 2.1.5.2.14 Steuerung für das Applet: Die superx.properties

Das SuperX-Applet greift u.a. auf eine Datei **superx.properties** zu , um zu erfahren, mit welchem Datenbanksystem gearbeitet wird (Informix/Postgres).

Für diese Datei gibt es im Kernmodul ein Muster
`$SUPERX_DIR/webserver/tomcat/webapps/superx/applet/superx.properties.sam`
das Sie nach `superx.properties` kopieren und wie folgt anpassen:

Die Adresse des Servlets wird normalerweise automatisch ermittelt, bei Netzwerkproblemen kann sie jedoch auch fest angegeben werden, dazu # vor SxServerURL entfernen und localhost ggfs. durch IP-Nummer/Rechnername ersetzen.

In der SuperX-Properties wird außerdem das Datenbanksystem (Variable `SxDB`) festgelegt, sowie das Logging (Variable `logToKonsole`). Bei der Installation von SuperX sollten Sie das Logging auf "all" setzen, im Echtbetrieb sollten Sie das Logging wie beim Servlet auf "none" setzen.

**Ein Beispiel für die superx. properties:**

```
#
# Die Adresse des Servlets wird normalerweise auto-
matisch ermittelt,
# bei Problemen kann sie hier fest angegeben wer-
den, dazu # vor SxServerURL entfernen und localhost
ggfs. durch IP-Nummer/Rechnername ersetzen
#SxServerURL=http://IP:8080/superx/servlet/Super-
XDBServlet
# Der Logging-Level logToKonsole kann eingeschaltet
werden: "none","errors"=fehler,"all"=alles)
logToKonsole=errors
#Wird im Applet in Titelleiste angezeigt
SxTitle=Testhochschule
#Das verwendete DB-System, möglich sind "Informix"
und "Postgres"
SxDB=Postgres
# HTML-Format Parameterleiste, Feldname in BOLD
SxParamStart=<html><body BGCOLOR=\"#ffffff\"><font
face=arial,helvetica size=-2>
SxParamEnd=</font></body></html>
SxParamBoldOn=<b>
SxParamBoldOff=</b>
```

Wenn der Webserver mehrere Mandanten in unterschiedlichen Datenbank bedient, muss es für jeden Mandanten eine superx.properties geben, die den zusätzlichen Parameter MandantenID enthält, z.B.

`MandantenID=7200` (vergl. entsprechendes Kapitel).

Schließlich muss man noch darauf achten, dass ggfs. der Tomcat-Port in der Firewall (standardmäßig Port 8080, evtl. noch 8007 und 8443) freigegeben ist.

Die Homepage von SuperX liegt standardmäßig unter
`http://<IP-Nummer des Servers>:8080/superx/`

#### 2.1.5.2.15Steuerung des XML-Frontends: PageComponents

Das XML-Frontend generiert aus XML-Datenströmen die Oberfläche im html-Format. So lässt sich die Oberfläche von SuperX beliebig mit XSLT anpassen, Details dazu finden Sie im SuperX-Entwicklerhandbuch.

**Vorbemerkung** Achtung: eine Änderung dieser Parameter ist **nur in SuperX-Releases** möglich. Wenn Sie SuperX als Teil von Edustore in HISinOne nutzen, sind die im Folgenden beschriebenen Maßnahmen nich tmöglich bzw. führen zu unvorhersehbaren Ergebnissen.

Das XML-Frontend mit DHTML-Techniken erlaubt es, wahlweise den Themenbaum als Javascript-"Baum" anzuzeigen (SuperX Kernmodul ab Version 3.0rc7-3.5rc2) oder als normales html-Menü (SuperX Kernmodul bis Version 3.0rc6 oder ab Version 4.0). Wenn der Javascript-Baum nicht genutzt werden soll, kann dieser wie folgt ein/ausgeschaltet werden:

**Abschaltung des Javascript-Baums**

Editieren Sie die Datei

```
$SUPERX_DIR/webserver/tomcat/webapps/superx/xml/pageCom-ponents_html_final.xsl
```

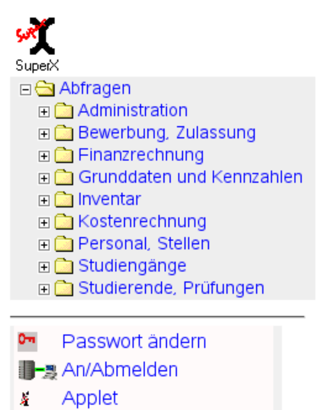
und setzen Sie folgende Anweisung aktiv:

```
<xsl:template name="showJavascriptMenue" >
<xsl:text>false</xsl:text>
</xsl:template>
```

Nach einem Tomcat-Neustart wird das html-Menü angezeigt.

Analog können Sie das Javascript-Menü einschalten:

Aktivierung des Javascript-Baums



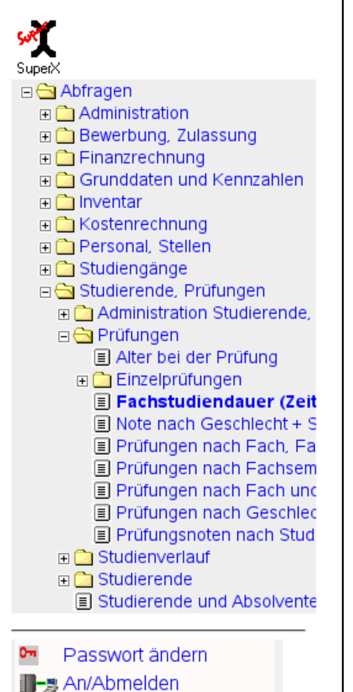

Editieren Sie die Datei

```
$SUPERX_DIR/webserver/tomcat/webapps/superx/xml/pageCom-ponents_html_final.xsl
```

und setzen Sie folgende Anweisung aktiv:

```
<xsl:template name="showJavascriptMenue" >
<xsl:text>true</xsl:text>
</xsl:template>
```

Aktivierung der Links zu den Masken im Javascript-Baum



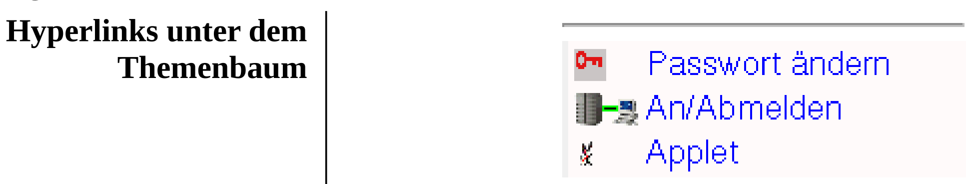

Wenn Sie darüber hinaus auch wollen, dass nicht nur die Themen, sondern auch die Masken im linken Menü angezeigt werden, setzen Sie eine weitere Variable auf "true":

```
<xsl:template name="showThemenbaumMask" >
<xsl:text>true</xsl:text>
</xsl:template>
```

Damit erscheinen Links auch die Masken.

Viele Hochschulen, die SuperX mit LDAP-Anmeldung nutzen oder anderweitig konfigurieren, wollen weitere Steuerungmöglichkeiten über das Aussehen des Menüframes nutzen. Sie können auch steuern, wie der Fuss des linken Navigationsframes aussehen soll; standardmäßig werden folgende Links angezeigt:

**Hyperlinks unter dem Themenbaum**




Editieren Sie Ihre Datei "pageComponents_html_final.xsl":

Fügen Sie die folgenden Einträge in pageComponents_html_final.xsl vor dem Tag am Ende </xsl:stylesheet> ein:

**Ausblenden des Passwort-Ändern-Links:**

```
<xsl:template name="showPasswordChangeLink" >
<xsl:text>false</xsl:text>
</xsl:template>
```

**Ausblenden des Login/Logout-Links:**

```
<xsl:template name="showLogoutLink" >
<xsl:text>false</xsl:text>
</xsl:template>
```

**Ausblenden des Applet-Links:**

```
<xsl:template name="showAppletLink">
<xsl:text>false</xsl:text></xsl:template>
```

Sie können unter diesem Bereich noch weitere HTML-Elemente einbauen. Dafür gibt es ein in der Auslieferung von SuperX befindliches leeres Template menue_fuss:

**Einblenden weiterer Links oder Texte: Beispiel**

```
<xsl:template name="menue_fuss">
<p>SuperX an der Universität xy</p></xsl:template>
```

Dies können Sie in pageComponents_html_final.xsl mit beliebigen Elementen füllen.

#### 2.1.5.2.16Einrichtung des Webservers bei mehreren Mandanten.

Wenn die Servlet-Engine mehrere SuperX-Mandanten in unterschiedlichen Datenbanken bedienen soll, muss es unter $SUPERX_DIR/webserver/tomcat/webapps/superx/WEB_INF eine Datei mandanten.cfg geben.

Darin müssen die MandantenIDs (typischerweise Hochschulnummern) der einzelnen Mandanten aufgeführt sein. (Jeweils eine ID pro Zeile). Zusätzlich muss es dann nicht eine db.properties geben, sondern für jeden Mandanten eine nach dem Schema db_XXXX.properties, wobei XXXX für die MandantenID steht.

Bearbeiten mit dem propadmin können Sie die einzelnen db_XXXX.properties Dateien, indem Sie ins Verzeichnis $SUPERX_DIR/webserver/tomcat/webapps/superx/WEB_INF wechseln und dann den propadmin[12] starten mit

```
propadmin.x ./db_XXXX.properties.
```

Nach dem Start von Tomcat können Sie in den Logdateien (meist catalina.out oder localhost.log) kontrollieren, ob für jeden Mandanten ein Datenbank-ConnectionPool aufgebaut wurde.

---

[12] Im letzten Release waren versehentlich noch veraltete eine veraltete propadmin.x und propadmin.bat im Verzeichnis $SUPERX_DIR/webserver/tomcat/webapps/superx/WEB-INF vorhanden. Falls der Fehler NoClassDefFound auftaucht, löschen Sie diese zwei Dateien.

Unter `$SUPERX_DIR/webserver/tomcat/webapps/superx` sollte es für jeden Mandanten ein Unterverzeichnis mit dem Namen der MandantenID geben.

z.B.

```
$SUPERX_DIR/webserver/tomcat/webapps/superx/7200
$SUPERX_DIR/webserver/tomcat/webapps/superx/7300
$SUPERX_DIR/webserver/tomcat/webapps/superx/7400
```

In jedes der Mandantenunterverzeichnisse müssen einige Dateien und Verzeichnisse reinkopiert werden, da Tomcat mit symbolischen Links Probleme hat.

Rufen Sie im Verzeichnis
`$SUPERX_DIR/webserver/tomcat/webapps/superx`

das Skript `copytoMandantenDir.x MANDANTENID` auf (z.B. `copytoMandantenDir.x 7200`).

Falls noch nicht vorhanden wird ein Unterverzeichnis 7200 angelegt und alle Dateien dort hinkopiert.

Wechseln Sie dann in das Mandantenverzeichnis 7200.

Im Unterverzeichnis applet muss die `superx.properties` angelegt werden, wie im vorherigen Abschnitt beschrieben. Zusätzlich muss die MandantenID in der `superx.properties` angegeben werden, z.B.

```
MandantenID=7200
```

Ebenso muss in Unterverzeichnis xml in der Datei `anmeldung.htm`, die MandantenID als versteckter Parameter mit übergeben werden, z.B.

```
<input type="hidden" name="MandantenID" value="7200">
```

Die einzelnen Mandanten können SuperX dann mit der Url

`http://rechnername:8080/superx/MANDANTENID`

aufrufen, z.B.

http://www.plgr-bw.de:8080/superx/7200

Wenn bestimmte Mandanten das Upload-Servlet zum Hochladen von Dateien per Browser nutzen sollen, muss die web.xml angepasst werden, siehe dazu im Abschnitt zu Upload-Funktion den Punkt **Anpassung der web.xml**

### 2.1.5.2.17 Einrichtung von DBFORMS bei mehreren Mandanten

Wenn auch die Administrationsabfragen von DBFORMS genutzt werden sollen, müssen die Datenbankverbindungen in zwei Steuerungsdateien eingetragen werden: der `server.xml` für die Datenbankverbindung, und der `dbforms-config.xml` für die dbforms-Anbindung.

Die Vorbereitung des Tomcat-Servers für den Einsatz von dbforms wurde im Abschnitt zur server.xml erläutert. Für den Einsatz mehrerer Mandanten müssen die Datenquellen in der Datei `$SUPERX_DIR/webserver/tomcat/conf/server.xml` eingetragen werden.

Diese sähe dann z.B. für die Mandanten `7200` und `7300`, deren Datenbanknamen unter Postgres `pg7200` und `pg7300` lauten, so aus:

**Der mandantefähige SuperX-Kontext in der server.xml**

```
<Context path="/superx" docBase="superx" debug="0"
                 reloadable="true" crossContext="true">
          <Logger className="org.apache.catalina.logger.FileLogger"
                     prefix="localhost_superx_log." suffix=".txt"
              timestamp="true"/>
         <Environment name="maxExemptions" type="java.lang.Integer"
                      value="15"/>
          <Parameter name="context.param.name" value="context.param.-
value"
                     override="false"/>
          <Resource name="jdbc/mandant_7200" auth="Container"
          type="javax.sql.DataSource"/>
          <Resource name="jdbc/mandant_7300" auth="Container"
          type="javax.sql.DataSource"/>
<!--Mandant 7200 wird angebunden:
<ResourceParams name="jdbc/mandant_7200">
  <parameter>
    <name>factory</name>
    <value>org.apache.commons.dbcp.BasicDataSourceFactory</value>
  </parameter>
  <parameter>
    <name>driverClassName</name>
    <value>org.postgresql.Driver</value>
  </parameter>
  <parameter>
    <name>url</name>
    <value>jdbc:postgresql://localhost/pg7200</value>
  </parameter>
  <parameter>
    <name>username</name>
    <value>superx</value>
  </parameter>
  <parameter>
    <name>password</name>
    <value>anfang12</value>
  </parameter>
  <parameter>
    <name>maxActive</name>
    <value>7</value>
  </parameter>
  <parameter>
    <name>maxIdle</name>
    <value>5</value>
  </parameter>
  <parameter>
    <name>maxWait</name>
    <value>-1</value>
  </parameter>

  <parameter>
<name>removeAbandoned</name>
<value>true</value>
 </parameter>
<parameter>
<name>removeAbandonedTimeout</name>
<value>10</value>
</parameter>
</ResourceParams>
```

```
<!--Mandant 7300 wird angebunden:
<ResourceParams name="jdbc/mandant_7300">
  <parameter>
    <name>factory</name>
    <value>org.apache.commons.dbcp.BasicDataSourceFactory</value>
  </parameter>
  <parameter>
    <name>driverClassName</name>
    <value>org.postgresql.Driver</value>
  </parameter>
  <parameter>
    <name>url</name>
    <value>jdbc:postgresql://localhost/pg7300</value>
  </parameter>
  <parameter>
    <name>username</name>
    <value>superx</value>
  </parameter>
  <parameter>
    <name>password</name>
    <value>anfang12</value>
  </parameter>
  <parameter>
    <name>maxActive</name>
    <value>7</value>
  </parameter>
  <parameter>
    <name>maxIdle</name>
    <value>5</value>
  </parameter>
  <parameter>
    <name>maxWait</name>
    <value>-1</value>
  </parameter>

  <parameter>
<name>removeAbandoned</name>
<value>true</value>
 </parameter>
<parameter>
<name>removeAbandonedTimeout</name>
<value>10</value>
</parameter>
</ResourceParams>
</Context>
```

Es werden also die Tags **<Resource… />** und **<ResourceParams… />** für jeden Mandanten dupliziert und konfiguriert. Jeder Mandanten-Datenquelle ist dann für dbforms über den Namen der Ressource ansprechbar, also in diesem Beispiel `7200` und `7300`.

Diese Datenquellen müssen dann wie im Abschnitt zu dbforms erläutert in der Datei `$SUPERX_DIR/webserver/tomcat/webapps/superx/WEB-INF/dbforms-config.xml` beim Tag **<dbConnection …/>** eingetragen werden:

**Das Ende der Datei dbforms-config.xml bei mandantenfähigem Betrieb**

```
...
<!--Hier endet Moduldefinition-->
<dbconnection id ="7200" isJndi="true"
name="java:/comp/env/jdbc/mandant_7200"/>
<dbconnection id="7300" isJndi="true"
name="java:/comp/env/jdbc/ mandant_7300"/>
</dbforms-config>
```

Wichtig ist, dass die Mandanten-ID mit der in der `mandanten.cfg` (s.o.) übereinstimmt.

### 2.1.5.3 Integration von Tomcat mit dem Apache

In Systemumgebungen, in denen bereits ein Webserver wie Apache läuft, bietet es sich an, den SuperX-Tomcat mit dem Webserver zu verbinden. Der Webserver kann so konfiguriert werden, dass Aufrufe zu `http://<<Servername>>/superx` direkt an Tomcat weitergeleitet werden können. Dies hat auch den Vorteil, dass die ungewöhnlichen Ports von Tomcat (8080 bzw. 8443) nicht in der Firewall freigegeben werden müssen. Außerdem kann die Verschlüsselung vom Apache durchgeführt werden, und es kann ein Load-Balaning eingeführt werden (Lastausgleich zwischen 2 Tomcat-Servern, gesteuert vom Apache). Wir empfehlen daher generell die Anbindung von Tomcat an den Apache für einen Produktivbetrieb.

#### 2.1.5.3.1 Installation des Apache-Tomcat-Connectors

Der Apache-Tomcat-Connector für Apache 1.3.x und Apache 2.x heißt `mod_jk` und ist ein Apache-Modul, das via DSO in eine vorhandene Apache-Installation gelinkt werden kann. Das `mod_jk` kann man herunterladen z.B. von http://jakarta.apache.org (im Downloadbereich unter Sources, die aktuelle Version ist 1.2.6). Unter SuSE Linux 8.2 und höher befindet sich das `mod_jk` im Paket `apache-tomcat-connectors`. Ein unter SuSE Linux 9.0 für den Apache 1.3.28 kompiliertes `mod_jk` liegt im Kernmodul unter `$SUPERX_DIR/webserver/apache/lib`), die Quellen liegen in `$SUPERX_DIR/webserver/apache/src/jakarta-tomcat-connectors-jk-1.2-src-current.tar.gz`.

Bei manchen Systemen ist es sinnvoll, den `mod_jk` selbst zu kompilieren. Zunächst müssen der Apache 1.3.x bzw. 2.x und das apxs-Tool installiert sein (apxs ist unter SuSE Linux Teil der *devel*-Package für Apache). Nun entpackt man die mod_jk-Quellen z.B. im Verzeichnis `/usr/src/apache/`

Dann geht man als root in das Verzeichnis
`/usr/src/apache/jakarta-tomcat-connectors-jk-1.2.6-src/jk/native`
und gibt ein (`$JAVA_HOME` und `/usr/sbin/apxs` müssen ggf. angepasst werden):

Unter Apache 1.3.x:

```
./configure --with-java-home=$JAVA_HOME --enable-EAPI --with-apxs=/usr/sbin/apxs
```

Unter Apache 2.x:

```
./configure --with-java-home=$JAVA_HOME --with-apxs=/usr/sbin/apxs2
```

Dann gilt für beide:

```
make
```

Nutzer von RedHat 9.x beachten bitte folgende Fußnote[13].

[13] Bei RedHat 9.x kann es Probleme geben. Wenn Sie die Fehlermeldung "`make[1]: *** [mod_jk.la] Error 1`" erhalten, müssen Sie den Aufruf wie folgt ändern:

```
make LIBTOOL=/etc/httpd/build/libtool
```

Danach ist das Modul kompiliert und wird in das Modulverzeichnis des Apache kopiert (z.B. `/usr/lib/apache`). Für Apache 1.3:

```
cp ./apache-1.3/mod_jk.so /usr/lib/apache
```

bzw. für Apache 2.x:

```
cp ./apache-2.0/mod_jk.so /usr/lib/apache2
```

Danach kann die Konfiguration des `mod_jk` beginnen, was im folgenden Kapitel beschrieben ist.

#### 2.1.5.3.2Umleitung von Requests vom Apache zu Tomcat

Die Konfiguration des Apache zur Anbindung an Tomcat ist im `tomcat-apache-howto` dokumentiert, der sich in jeder Download-Version des offiziellen Tomcat  befindet (webapps/doc).

Die Umleitung von Requests vom Apache zum Tomcat kann auch auf zwei Rechnern geschehen, z.B. um den Apache-Server in der DMZ und den Tomcat-Server im Intranet zu betreiben. Wir empfehlen letzteres aus Sicherheitsgründen, beachten Sie aber dabei, dass auch die Verbindung vom Apache-Server zum Tomcat via mod_jk verschlüsselt wird, z.B. über einen ssh-Tunnel.

In der SuperX-Distribution sind die Tomcat-spezifischen Dateien für die Anbindung an den Apache 1.3.x via `mod_jk` bereits enthalten, es müssen lediglich ein paar Anpassungen gemacht werden:

- Teil der SuperX-Distribution ist ein Konfigurationsbeispiel mit dem Namen `$SUPERX_DIR/webserver/tomcat/conf/superx_mod_jk.conf.sam`. dieses können Sie umbenennen nach superx_mod_jk.conf, und in der Datei den Pfad für das mod_jk-Modul (`mod_jk.so`) anpassen. Ausserdem kann der Logging-Level festgelegt werden (Werte: "debug", "warning", "error", im Echtbetrieb empfehlen wir "error").
- Bei manchen mod_jk- oder Apache Versionen muss man noch die Zeile
  JkMountCopy All
  hinzufügen, siehe Hyperlink.
- In der Datei `$SUPERX_DIR/webserver/tomcat/conf/workers.properties` muss der Parameter
  `workers.tomcat_home`
  auf den richtigen Pfad gesetzt werden (wenn Sie SuperX in `/home/superx` installiert haben, brauchen Sie hier keine Änderungen vornehmen).
  Ausserdem muss der richtige Pfad für `workers.java_home` gesetzt werden, sowie der Pfad-Demiliter `ps` für das Betriebssystem ("/" für Unix, "\" für Win, ":" für Mac)
  Wenn Sie den Apache auf einem separaten Rechner betrieben, dann müssen Sie beim Parameter `worker.ajp13.host` nicht "localhost", sondern den Rechnernamen / IP-Nr. des Tomcat-Servers eintragen.
- Danach fügen Sie am Ende der Apache-Konfigurationsdatei (unter SuSE Linux z.B. `/etc/httpd/httpd.conf`) die Zeile
  `Include /home/superx/webserver/tomcat/conf/superx_mod_jk.conf)`
  ein. Außerdem müssen Sie ggf. die Umgebungsvariable DirectoryIndex auf `index.htm` setzen (nicht nur index.html), da die Startseite in den Verzeichnissen immer `index.htm` heißt.
  Danach starten Sie Apache neu (`apachectl restart`).
- In der Datei `$SUPERX_DIR/webserver/tomcat/conf/server.xml` auf dem Tomcat-Rechner kann dann der http-Connector 8080 auskommentiert werden, und der Apache-Connector `ajp13` kann benutzt werden; standardmäßig geht dieser über den Port 8009. Dieser Connector ist bei Auslieferung von SuperX aktiviert.
- Dann starten Sie zuerst Tomcat neu, und dann den Apache. Danach müsste auf dem Webserver das Verzeichnis `superx` gemounted sein, und alle Anfragen mit der Endung *.jsp bzw. in das servlet-Verzeichnis gehen zu Tomcat.

- Danach müssen Sie ggf. in der Datei
  `$SUPERX_DIR/webserver/tomcat/webapps/superx/applet/superx.properties`
  den Port für Tomcat (Vorbelegung: ":8080") rausnehmen (mit "#" auskommentieren).

Sobald Tomcat so an den Apache angebunden ist, kann auch die Verschlüsselung über den Apache laufen. Wenn das Apache-Modul `mod_ssl` installiert und konfiguriert ist, dann werden auch automatisch anhand des Präfixes `http://..` und `https://...` Anfragen an Tomcat weitergeleitet.

#### 2.1.5.3.3Einrichtung von Load Balancing

Mit dem eingerichteten `mod_jk` lässt sich recht einfach ein Lastausgleich zwischen mehreren Tomcats oder eine Trennung von Apache und Tomcat auf zwei Server implementieren. Die Konfiguration findet statt im Tomcat-Konfigurationsverzeichnis `$SUPERX_DIR/webserver/tomcat/conf`.

Dazu muss in der Datei `workers.properties` ein zweiter Worker eingerichtet werden, siehe Beispieldatei `workers.properties.lb.sam` in der SuperX-Distribution. Außerdem muss in der `server.xml` auf dem Rechner, auf dem der Apache läuft, ein zweiter AJP-Connector eingerichtet werden, z.B. am Port 8010.

**Auszug aus der server.xml des Apache-Rechners**

```
<!-- Define an AJP 1.3 Connector on port 8009 -->
<Connector className="org.apache.ajp.tomcat4.Ajp13Connector"
              port="8009" minProcessors="5"
maxProcessors="75"
               acceptCount="10" debug="0"/>
<Connector className="org.apache.ajp.tomcat4.Ajp13Connector"
              port="8010" minProcessors="5"
maxProcessors="75"
               acceptCount="10" debug="0"/>
```

Der AJP-Connector am Port 8010 muss dann auf dem zweiten Tomcat-Server eingetragen werden (nur dieser, nicht der 8009er).

Danach kann in der mod_jk-Konfigurationsdatei ein Lastausgleich eingerichtet werden (siehe Beispieldatei `superx_mod_jk_lb.conf.sam`).

#### 2.1.5.3.4Einrichten von SSL beim Apache 1.3.x unter Linux

Der Apache Version 1.3.x benötigt für den SSL-Betrieb das Modul `mod_ssl`[14], im Apache 2.x ist das SSL-Modul bereits Bestandteil des "Kern"-Apache.

Wenn Sie Apache2 einsetzen, blättern Sie bitte weiter.

Mit Hilfe des Openssl-Paketes sowie können Schlüssel für den Server erzeugt werden. Im Folgenden erläutern wir das Vorgehen unter SuSE Linux 8-9, für andere Distributionen müssen Sie ggf. die Verzeichnisnamen anpassen. Für die Installation verwenden wir zunächst ein selbst signiertes Zertifikat, was zwar den Nachteil hat, dass die Anwender vor dem Aufruf der Webseite eine Warnung erhalten ("Diese Seite stammt aus einer nicht vertrauenswürdigen Quelle..."), der Vorteil ist aber, dass das Vorgehen relativ ein-

---
[14] http://www.modssl.org

heitlich ist und später bei Bedarf leicht um ein öffentliches Zertifikat erweitert werden kann. Wenn die Verschlüsselung mit einem selbst signierten Zertifikat funktioniert, dann ist der Rest relativ einfach.

Wir führen alle Schritte als user `root` durch, und gehen z.B. davon aus, dass wir uns im Verzeichnis `/root` befinden.

Zunächst muss ein Zertifikat erzeugt werden (bitte passen die die Verzeichnisnamen jeweils an Ihr OpenSSL-Paket an):

```
/usr/share/ssl/misc/CA.sh -newca
```

Sie geben ein Passwort ein und die jeweiligen Angaben (Land, Organisation etc.). Beim "Common Name" muss der DNS-Servername angegeben werden.

Das öffentliche CA-Zertifikat liegt nun in `/root/demoCA/cacert.pem` und der private Schlüssel liegt in `/root/demoCA/private/cakey.pem`.

Wenn Sie Ihren Besuchern das CA-Zertifikat zum Download anbieten möchten, müssen Sie dieses zuerst in das entsprechende DER-Format konvertieren:

```
openssl x509 -in demoCA/cacert.pem -out capub.crt -outform DER
```

Es wird die Datei `/root/capub.crt` erzeugt.

Wenn Sie Ihr Zertifikat bei einer Zertifizierungstelle signieren lassen möchten, müssen Sie für den Server-Dienst `http` ein weiteres Zertifikat erzeugen. Mit

```
/usr/share/ssl/misc/CA.sh -newreq
```

erzeugen Sie ein neues Zertifikat, das Passwort sollte auf keinem Fall dem obigen Server-Zertifikat entsprechen. Dann müsen Sie die Zertifizierungsanfrage aus der Datei `newreq.pem` mit folgendem Befehl in eine separate Datei speichern. Senden Sie auf gar keinen Fall die Datei `newreq.pem` zur Zertifizierungsstelle, da diese zusätzlich Ihren privaten Schlüssel enthält.

```
openssl req -text -in newreq.pem -out request.pem
```

Wie und in welchem Format Sie die Anfrage an die von Ihnen ausgewählte Zertifizierungsstelle senden müssen, erfahren Sie von der entsprechenden Zertifizierungsstelle.

Mit dem weiter oben erstellten CA-Zertifikat können Sie Ihr http-Zertifikat folgendermaßen selbst signieren:

```
/usr/share/ssl/misc/CA.sh -sign
```

Es wird eine Datei `newcert.pem` erzeugt. Nachdem Sie nun ein signiertes Zertifikat für Ihre Anwendung erstellt haben, müssen Sie dieses nur noch in das entsprechende Verzeichnis kopieren und in der Konfigurationsdatei eintragen. Der Apache erwartet den privaten Schlüssel in einer separaten Datei, in solchen Fällen können Sie den privaten Schlüssel wie folgt extrahieren

```
openssl rsa -in newreq.pem -out newkey.pem
```

Nun bereiten wir den Neustart des Apache mit ssl-Modul vor. Die Einbindung mit Loadmodule und AddModule muss bei den meisten Distributionen nicht manuell gemacht werden.

**Apache 1.3.x-SSL-Verschlüsselung unter SuSE Linux**

Unter SuSe Linux müssen Sie zunächst eine Umgebungsvariable setzen. Schreiben Sie in der Datei `/etc/sysconfig/apache`:
`HTTPD_SEC_MOD_SSL=yes`
Und starten Sie danach einmal das Script `SuSEconfig`.

Im Apache muss nun in der Steuerungsdatei `httpd.conf` der Pfad zum privaten und öffentlichen Schlüssel angegeben werden. Das folgende Beispiel geht davon aus, dass der öffentliche CA-Schlüssel auf der Website des Users superx (Modul `public_html` des Apache) unter `/home/superx/public_html` steht, und dass der private Schlüssel des Servers vom User `root` im Verzeichnis `/root/demoCA` erzeugt wurde.

**Auszug aus der Apache-Konfigurationsdatei httpd.conf**

```
<VirtualHost <<Ipnr. des Webservers>>:443>
  ServerName <<Ihr DNS-Servername>>
  #   SSL Engine Switch:
  #   Enable/Disable SSL for this virtual host.
  SSLEngine on
  #   SSL Cipher Suite:
  SSLCipherSuite ALL:!ADH:!EXPORT56:RC4+RSA:+HIGH:+MEDIUM:+LOW:
+SSLv2:+EXP:+eNULL
  #   Server Certificate:
  SSLCertificateFile /home/superx/public_html/capub.crt
  #   Server Private Key:
  SSLCertificateKeyFile /root/demoCA/private/cakey.pem
```

Danach müssen Sie in `/etc/sysconfig/apache` die Systemvariable HTTPD_START_TIMEOUT auf einen sinnvollen Wert setzen, z.B. 10. Sie haben dann beim Start des Apache 10 Sek. Zeit, dass CA-Passwort einzugeben.

Wenn sie wünschen, dass der Apache beim Booten ohne Passwort-Abfrage startet, dann müssen Sie das CA-Passwort löschen und die Leserechte für den privaten Schlüssel ändern (nur root und der Apache-Daemon haben Leserecht)[15]. Dies ist allerdings ein Sicherheitsrisiko; der Server wird leichter kompromittierbar, wenn ein Hacker auf den Rechner kommt und die Datei lesen kann, kann er den Schlüssel missbrauchen. Unserer Erfahrung nach ist aber nur dieser Weg gangbar, denn bei einem Reboot nach Stromausfall würde der gesamte Webserver sonst nicht laufen!

Wir geben als root im Verzeichnis `/root/demoCA/private` ein:

```
openssl rsa -in cakey.pem -out cakey2.pem
```

(1x mit der Passphrase bestätigen).

Dann wird ein Schlüssel ohne Passphrase erzeugt. Wenn wir diesen dann wiederum in `/etc/httpd/httpd.conf` eintragen:

```
#SSLCertificateKeyFile /root/demoCA/private/cakey.pem
SSLCertificateKeyFile /root/demoCA/private/cakey2.pem
```

Dann startet der Apache ohne Passwortabfrage. In diesem Fall kann man auch die Variable `HTTPD_START_TIMEOUT` auf 1 zurücksetzen.

[15] Auch in der offiziellen Doku von `mod_ssl` wird dies empfohlen, mit dem Zusatz,dass nur root und der apache-daemon diese Datei lesen darf.
http://www.modssl.org/docs/2.0/#FAQ-nopass

Zum Abschluss können Sie bei einem selbst signierten Zertifikat die oben erstellte Datei /root/capub.-crt auf den Webserver kopieren und mit folgendem Link auf Ihrer Webseite verfügbar machen:

```
<a href="capub.crt" type="application/x-x509-ca-cert">CA-Zertifikat</a>
```

Die Anwender können dann mit Klick auf Link das Zertifikat importieren und somit im Browser speichern, so dass die Warnung, dass die Quelle nicht vertrauenswürdig ist, nicht mehr kommt. Wir haben auch den Eindruck, dass das Applet dann schneller arbeitet.

#### 2.1.5.3.5Einrichten von SSL beim Apache 2.x unter SuSE Linux

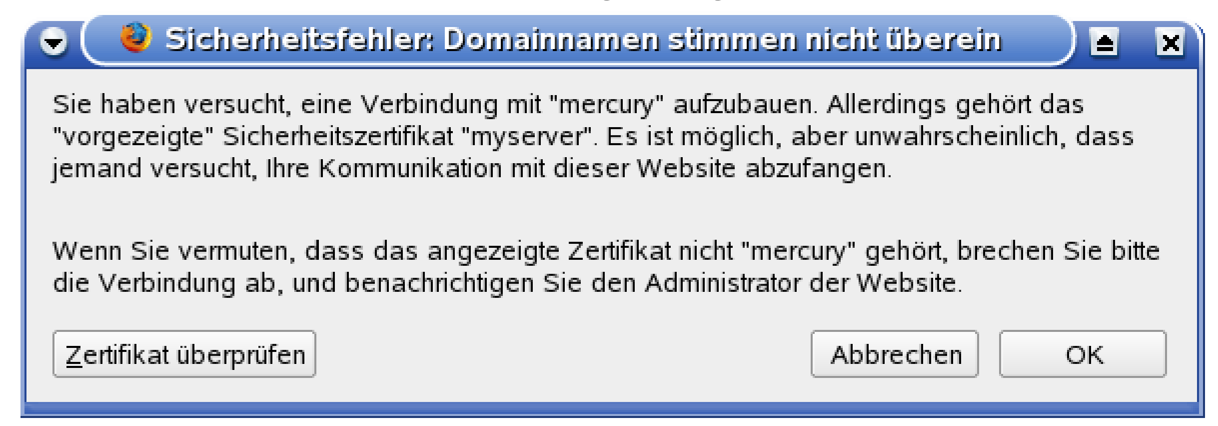
Das Modul **ssl** ist im Apache 2.x nicht mehr separat zu installieren, sondern bereits im Lieferumfang enthalten, das Modul muss nur in den entsprechenden LoadModule und Include-Abschnitten geladen werden.

Wir führen alle Schritte als user root durch, und gehen z.B. davon aus, dass wir uns im Verzeichnis /root befinden.

Zunächst muss ein Zertifikat erzeugt werden (bitte passen Sie die Verzeichnisnamen jeweils an Ihr OpenSSL-Paket an):

```
/usr/share/ssl/misc/CA.sh -newca
```

Sie geben ein Passwort ein und die jeweiligen Angaben (Land, Organisation etc.). Beim "Common Name" muss der DNS-Servername des Webservers angegeben werden - dies ist wichtig, denn sonst erhalten die Anwender beim Aufruf der Seit eine Warnung, die ungefähr so aussieht:




. Ein Challenge Passwort ist erst einmal nicht notwendig. Am Ende der Prozedur muss man noch einmal das eingegebene Passwort eingeben.

**Ein Beispiel**

```
mercury:~ # /usr/share/ssl/misc/CA.sh -newca
CA certificate filename (or enter to create)

Making CA certificate ...
Generating a 1024 bit RSA private key
.................................++++++
.............++++++
writing new private key to './demoCA/private/./cakey.pem'
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:DE
State or Province Name (full name) [Some-State]:NRW
Locality Name (eg, city) []:Wuppertal
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Memtext
Organizational Unit Name (eg, section) []:Workshop
Common Name (eg, YOUR name) []:192.168.0.108
Email Address []:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
Using configuration from /etc/ssl/openssl.cnf
Enter pass phrase for ./demoCA/private/./cakey.pem:
Check that the request matches the signature
Signature ok
Certificate Details:
        Serial Number: 0 (0x0)
        Validity
            Not Before: Mar 13 13:22:45 2007 GMT
            Not After : Mar 12 13:22:45 2010 GMT
        Subject:
...

Certificate is to be certified until Mar 12 13:22:45 2010 GMT (1095 days)

Write out database with 1 new entries
Data Base Updated
```

Das öffentliche CA-Zertifikat liegt nun in /root/demoCA/cacert.pem und der private Schlüssel liegt in /root/demoCA/private/cakey.pem..

Nun werden die Schlüssel dem Apache2 bekannt gemacht. Die einzelnen Konfigurationsparameter werden bei SuSE Linux über die Sysconfig gesetzt:

**Apache2 mit SSL unter SuSE Linux**

SuSE-typisch wird die Konfiguration in einer Datei im Verzeichnis /etc/sysconfig abgelegt, nämlich in apache2. Dort setzen Sie in der Direktive

```
APACHE_CONF_INCLUDE_FILES=
"/home/superx/webserver/tomcat/conf/superx_mod_jk.conf
/etc/apache2/vhosts.d/myhost-ssl.conf"
```

die Tomcat-Anbindung und den Virtuellen SSL-Host. Letzteren konfigurieren Sie am besten, indem Sie die Vorlage /etc/apache2/vhosts.d/vhost-ssl.template kopieren, z.b. wie oben nach myhost-ssl.conf.

Weiter unten in /etc/sysconfig/apache2 setzen Sie die Direktive

```
APACHE_SERVER_FLAGS="SSL"
```

Damit werden in verschiedenen anderen conf-Dateien die Abfragen <ifDefine SSL> positiv aufgelöst und die jeweiligen Direktiven darin werden aktiviert.

Nach dem Ändern der Datei /etc/sysconfig/apache2 müssen Sie als User root das Script SuSEconfig ausführen.

Bei anderen Linux-Distributionen entfällt die sysconfig. Auch unabhängig von der Distribution wird beim Apache2 nicht mehr die gesamte Konfiguration in einer großen httpd.conf gesammelt, sondern in separaten conf-Dateien. Bei virtuellen Hosts zum Beispiel befinden sich die Konfigurationen in Dateien mit der Endung *.conf im Verzeichnis vhosts.d. Der Startpunkt ist aber immer die httpd.conf (standardmäßig in /etc/apache2).

Wenn Sie keine Virtual Hosts nutzen, dann können Sie den Abschnitt, der im Konfigurationsbeispiel /etc/apache2/vhosts.d/vhost-ssl.template beschrieben ist auch in der Datei /etc/apache2/default-server.conf einfügen:

**/etc/apache2/default-server.conf**

```
##
## SSL Virtual Host Context
##

<VirtualHost _default_:443>

        #  General setup for the virtual host
        DocumentRoot "/srv/www/htdocs"
        ServerName 192.168.0.108:443
        #ServerAdmin webmaster@example.com
        ErrorLog /var/log/apache2/error_log
        TransferLog /var/log/apache2/access_log

        #   SSL Engine Switch:
        #   Enable/Disable SSL for this virtual host.
        SSLEngine on

        #   SSL Cipher Suite:
        #   List the ciphers that the client is permitted to ne-
gotiate.
        #   See the mod_ssl documentation for a complete list.
        SSLCipherSuite ALL:!ADH:!EXPORT56:RC4+RSA:+HIGH:+MEDIUM:
+LOW:+SSLv2:+EXP
:+eNULL

        #   Server Certificate:
        #   Point SSLCertificateFile at a PEM encoded certific-
ate.  If
        #   the certificate is encrypted, then you will be
prompted for a
        #   pass phrase.  Note that a kill -HUP will prompt
again.  Keep
        #   in mind that if you have both an RSA and a DSA cer-
tificate you
        #   can configure both in parallel (to also allow the
use of DSA
        #   ciphers, etc.)
        SSLCertificateFile /root/demoCA/cacert.pem
        #SSLCertificateFile /etc/apache2/ssl.crt/server.crt
        #SSLCertificateFile /etc/apache2/ssl.crt/server-dsa.crt

        #   Server Private Key:
        #   If the key is not combined with the certificate, use
this
        #   directive to point at the key file.  Keep in mind
that if
        #   you've both a RSA and a DSA private key you can con-
figure
        #   both in parallel (to also allow the use of DSA
ciphers, etc.)
        SSLCertificateKeyFile /root/demoCA/private/cakey.pem
        #SSLCertificateKeyFile /etc/apache2/ssl.key/server.key
        #SSLCertificateKeyFile /etc/apache2/ssl.key/server-
-dsa.key
...
```

Danach müssen Sie in `/etc/sysconfig/apache2` die Systemvariable HTTPD_START_TIMEOUT auf einen sinnvollen Wert setzen, z.B. 10. Danach wie immer `SuSEconfig` ausführen.

Sie haben dann beim Start des Apache 10 Sek. Zeit, dass CA-Passwort einzugeben.

Wenn sie wünschen, dass der Apache beim Booten ohne Passwort-Abfrage startet, dann müssen Sie das CA-Passwort löschen und die Leserechte für den privaten Schlüssel ändern (nur root und der Apa-

che-Daemon haben Leserecht)[16]. Dies ist allerdings ein Sicherheitsrisiko; der Server wird leichter kompromittierbar, wenn ein Hacker auf den Rechner kommt und die Datei lesen kann, kann er den Schlüssel missbrauchen. Unserer Erfahrung nach ist aber nur dieser Weg gangbar, denn bei einem Reboot nach Stromausfall würde der gesamte Webserver sonst nicht laufen!

Wir geben als root im Verzeichnis /root/demoCA/private ein:

```
openssl rsa -in cakey.pem -out cakey2.pem
```

(1x mit der Passphrase bestätigen).

Dann wird ein Schlüssel ohne Passphrase erzeugt. Wenn wir diesen dann wiederum in /etc/httpd/httpd.conf eintragen:

```
#SSLCertificateKeyFile /root/demoCA/private/cakey.pem
SSLCertificateKeyFile /root/demoCA/private/cakey2.pem
```

Dann startet der Apache ohne Passwortabfrage. In diesem Fall kann man auch die Variable HTTPD_START_TIMEOUT auf 1 zurücksetzen.

Wenn Sie Ihren Besuchern das öffentliche CA-Zertifikat zum Download anbieten möchten, müssen Sie dieses zuerst in das entsprechende DER-Format konvertieren:

```
openssl x509 -in demoCA/cacert.pem -out capub.crt -outform DER
```

Es wird die Datei /root/capub.crt erzeugt. Auf diese Datei wird in der Apache-Variable SSLCertificateFile verwiesen (statt wie oben auf /root/demoCA/cacert.pem)

```
#   Server Certificate:
        #   Point SSLCertificateFile at a PEM encoded certific-
ate.  If
        #   the certificate is encrypted, then you will be
prompted for a
        #   pass phrase.  Note that a kill -HUP will prompt
again.  Keep
        #   in mind that if you have both an RSA and a DSA cer-
tificate you
        #   can configure both in parallel (to also allow the
use of DSA
        #   ciphers, etc.)
        SSLCertificateFile /root/capub.crt
        #SSLCertificateFile /etc/apache2/ssl.crt/server.crt
        #SSLCertificateFile /etc/apache2/ssl.crt/server-dsa.crt
```

Wenn Sie Ihr Zertifikat bei einer Zertifizierungstelle signieren lassen möchten, müssen Sie die Zertifizierungsanfrage erzeugen. Mit

```
/usr/share/ssl/misc/CA.sh -newreq
```

erzeugen Sie ein neues Zertifikat, das Passwort sollte auf keinem Fall dem obigen Server-Zertifikat entsprechen.

[16] Auch in der offiziellen Doku von mod_ssl wird dies empfohlen, mit dem Zusatz,dass nur root und der apache-daemon diese Datei lesen darf.
http://www.modssl.org/docs/2.0/#FAQ-nopass

Danach müssen Sie die Datei `newreq.pem` mit folgendem Befehl in eine separate Datei speichern. Senden Sie auf gar keinen Fall die Datei `newreq.pem` zur Zertifizierungsstelle, da diese zusätzlich Ihren privaten Schlüssel enthält.

```
openssl req -text -in newreq.pem -out request.pem
```

Wie und in welchem Format Sie die Anfrage an die von Ihnen ausgewählte Zertifizierungsstelle senden müssen, erfahren Sie von der entsprechenden Zertifizierungsstelle.

Mit dem weiter oben erstellten CA-Zertifikat können Sie Ihr http-Zertifikat folgendermaßen selbst signieren:

```
/usr/share/ssl/misc/CA.sh -sign
```

Es wird eine Datei `newcert.pem` erzeugt. Nachdem Sie nun ein signiertes Zertifikat für Ihre Anwendung erstellt haben, müssen Sie dieses nur noch in das entsprechende Verzeichnis kopieren und in der Konfigurationsdatei eintragen. Der Apache erwartet den privaten Schlüssel in einer separaten Datei, in solchen Fällen können Sie den privaten Schlüssel wie folgt extrahieren

```
openssl rsa -in newreq.pem -out newkey.pem
```

### 2.1.6Anpassungen auf den Client-Rechnern

Der Vorteil von browser-basierten Webclients ist es, dass prinzipiell keine Installationen auf den Clients notwendig sind, und dass sie plattformübergreifend arbeiten. Nur für das SuperX-Applet muss man das Java-Plugin installiert haben.

#### 2.1.6.1Einstellungen für den Ajax-Client

Um mit dem Browser komfortabel arbeiten zu können, sollten wenn möglich die aktuellen, gebräuchlichen Browser eingesetzt werden, z.B. Firefox, Netscape, Seamonkey oder den Internet Explorer:

- Mozilla Firefox 1.5 oder höher, Mozilla 1.4 oder höher, Seamonkey 1.0 oder höher
- Internet Explorer 6.0 oder höher

Weiterhin ist es notwendig, dass die Anwender mit Bearbeitungszugriff auch Javascript einschalten (beim IE nennt sich dies "Active Scripting"). Man kann dies auch nur für bestimmte Server (bzw. Sicherheitszonen) tun, so dass Sie nur den Superx-Server freischalten müssen. Außerdem sollten Sie hier die Datenübermittlung zwischen Frames erlauben.

#### 2.1.6.2Installation der Java-Runtime

Das SuperX-Applet wird bei jedem Aufruf (je nach Cacheing des Browsers) neu geladen; der Umstieg auf neue Versionen des Applets ist also ohne lokale Installationen möglich. Eine Bedienungsanleitung zum Java-Client findet sich unter

`$SUPERX_DIR/doc/benutzerhandbuch_applet`

Für die Installation der Java-Runtime reicht es meist aus, zur Aufruf-Seite vom Applet zu surfen. Es wird

dann eine Installationsaufforderung inkl. Download von http://java.sun.com gestartet. Java von SUN, für andere Java-Versionen (IBM Java, GNU Java) wurden Probleme berichtet.

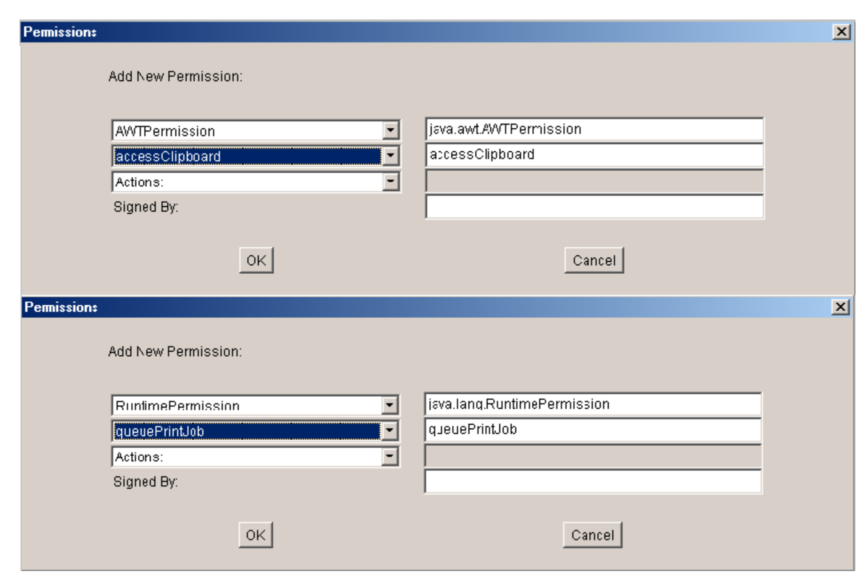
Für die Installation der Java Runtime benötigen Sie Administrationsrechte auf Ihrem Rechner.

#### 2.1.6.2.1Manuelle Anpassungen der Policy

Bei einigen Windows-Umgebungen (z.B. mit Netscape 6.1, ohne IE, oder mit Windows XP) läßt sich die Policy nicht scriptgesteuert installieren. Man muss dann die Policy dialogisch einrichten. Starten Sie dazu die Anwendung `policytool`, die sich im Lieferumfang der Java-Runtime befindet. Wenn Sie die Anwendung z.B. unter `C:\Programme\Java\JRE\1.6.1_02\bin\policytool.exe` installiert haben, dann starten Sie die Anwendung mit Doppelklick und gehen wie folgt vor:

Die AWT-Permission "AccessClipboard" muss gesetzt werden.

Die Runtime-Permission "queuePrintJob" muss gesetzt werden.




Danach klicken Sie auf "Done" und speichern die Policy im Home-Verzeichnis Ihrer Windows-Kennung, z.B. `c:\dokumente und einstellungen\<<Ihre Kennung>>\.java.policy`

#### 2.1.6.2.2Installation des Applets unter UNIX / Linux

Die Installationssite von SuperX erkennt, ob es sich um einen Linux-Browser handelt. Die Anwender werden zum Download auf die Seiten von Sun verwiesen.

Unter UNIX / Linux werden zunächst die Dateien der Java-Runtime bzw. des JDK 1.6.x der Firma SUN installiert (s.o.). In Mozilla 1.4 oder höher bzw. Netscape 6.x oder 7.x wird das SuperX-Applet am besten unterstützt.

Bei der Installation des Browsers ist zu beachten, daß im Plugins-Verzeichnis der Browserinstallation ein symbolischer Link auf die `libjavaplugin_oji.so` von der Java-Runtime gelegt wird, z.B. für Netscape

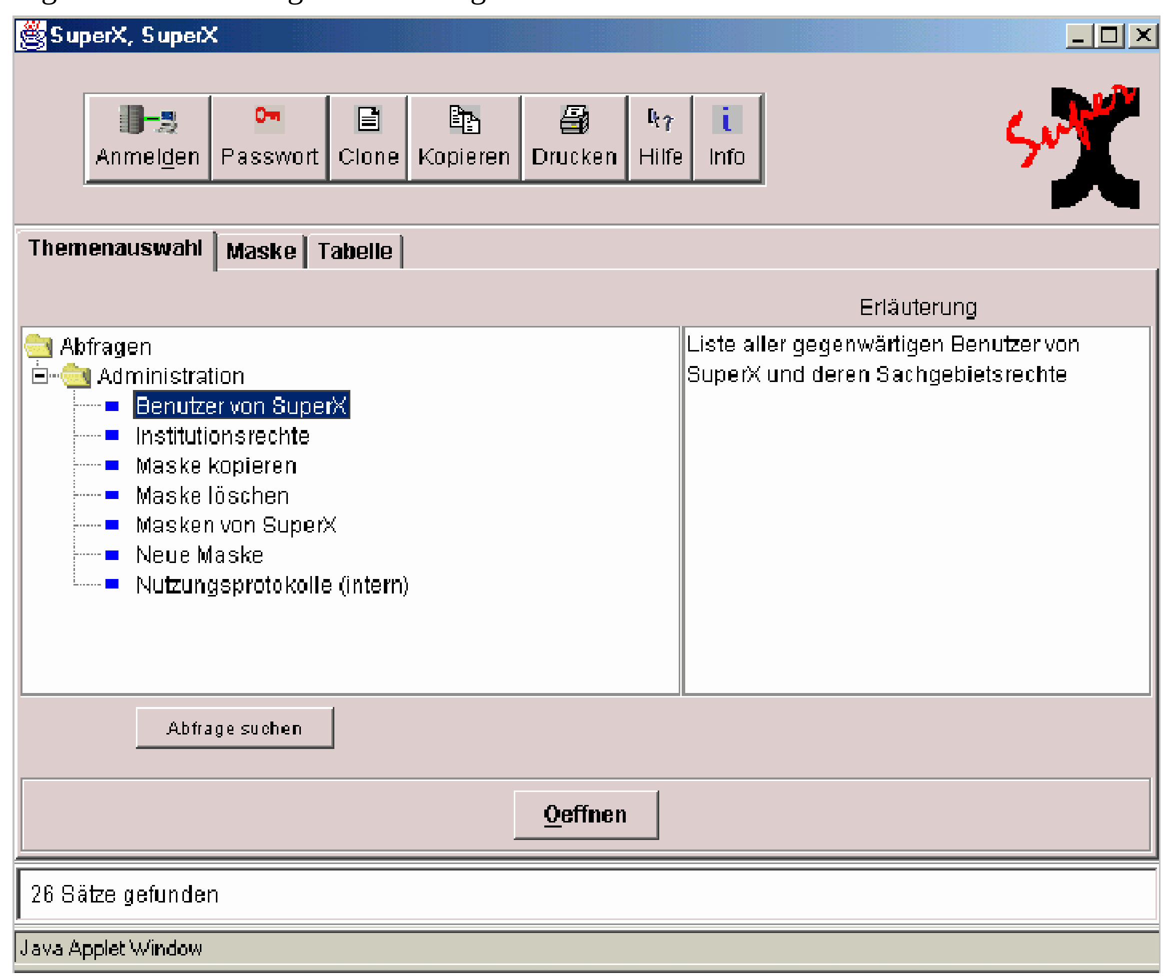
```
ln –s /usr/lib/java2/jre/plugin/i386/ns610/libjavaplugin_oji.so
/opt/netscape/plugins/libjavaplugin_oji.so
```

Bei Mozilla 1.x und Java 1.4.1 muss u.U. ein spezielles Java-1.4.1 verwendet werden, da das "normale" Java 1.4.1 mit einem anderen Compiler erzeugt wurde. Für Mozilla ist es zwingend notwendig, eine mit dem gcc 3.2.2 kompiliertes Paket zu verwenden, das auf der Website von Blackdown[17] zu beziehen ist. Bei Java 1.4.2 soll dieses Problem laut Aussage von SUN nicht mehr bestehen, da auch dieses mit dem gcc 3.2.2 kompiliert ist. Wir haben das Vorgehen bei einem vorinstallierten Java von Sun in SuSE Linux 8.x-9.0 getestet[18], hier gab es keine Probleme. Bei RedHat 9.0 klappte es allerdings nicht.

Nach der Installation von Java lautet der obige Befehl (wenn sowohl Java als auch Mozilla in `/usr/local` installiert sind):

```
ln –s /usr/local/jdk1.4.1/jre/plugin/i386/mozilla/javaplugin_oji.so /usr/local/mozilla/plugins/
```

Bei erfolgreicher Anmeldung erscheint folgendes Fenster:




### 2.1.6.3 Bei Problemen mit dem Start des Applets

Wenn es Probleme mit dem Start des Applets gibt, kann dies verschiedene Ursachen haben.

Unter Netscape ist aufgefallen, dass bei verschlüsselter Verbindung auf dem Server die Datei `$superx-dir/webserver/tomcat/webapps/superx/applet/superx_help/superx.hs` im gleichen Verzeichnis auch mit dem Namen `superx_de_DE.hs` existieren muss.

---

[17] www.blackdown.org

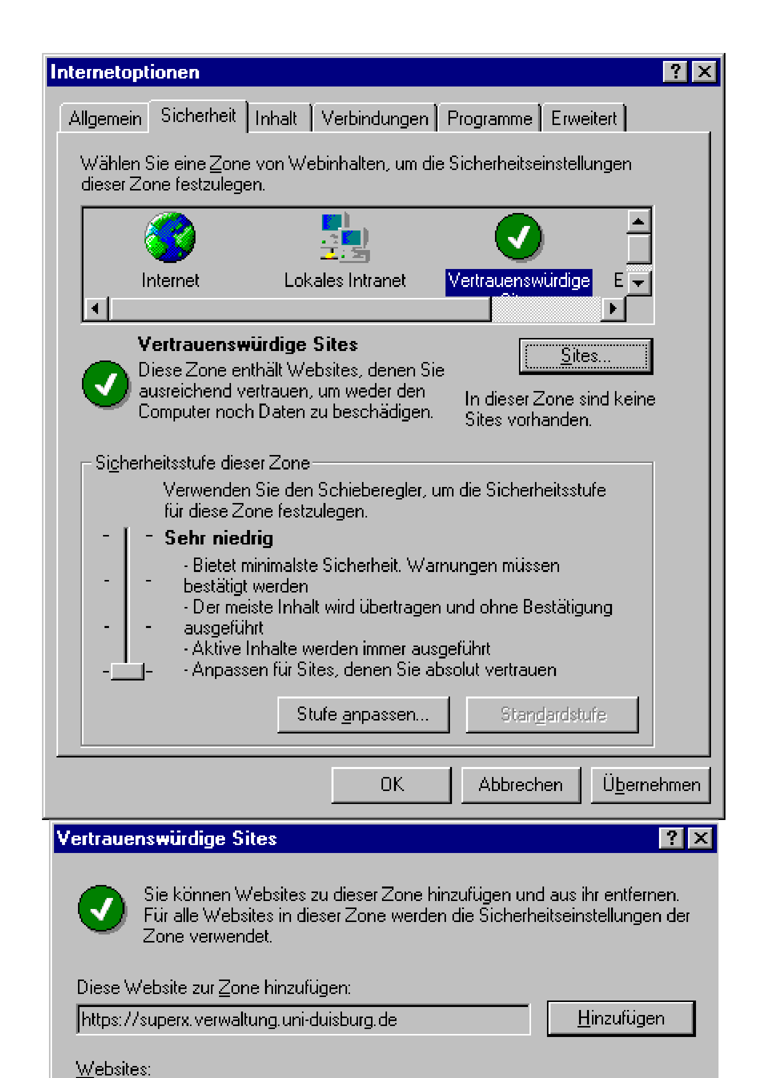
[18] Der Pfad des vorinstallierten Plugins mit der JRE lautet bei SuSE `/usr/lib/SunJava2/jre/plugin/i386/ns610-gcc32`

Eine weitere Ursache können **Sicherheitseinstellungen** sein. Fügen Sie Ihren SuperX-Server zur Liste der vertrauenswürdigen Sites hinzu.

Hier als Beispiel die Einstellung für den Duisburger SuperX-Server im InternetExplorer.

Im InternetExplorer und Extras / Internetoptionen, Registerkarte „Sicherheit“ Punkt Vertrauenswürdige Sites. Auf „Sites“ klicken.

Danach gibt man wie gezeigt den SuperX-Server ein und klickt auf Hinzufügen und OK.




Im **lokalen Netz** kann es durch den **Proxy** zu Problemen kommen. Man sollte daher den Proxy-Server für lokale Adressen umgehen.

Im InternetExplorer geht das folgendermaßen:

Zunächst wählt man im IE-Menü Extras-> Internetoptionen aus und wechselt zur Registerkarte Verbindungen. Dann klickt man auf LAN-Einstellungen.

Sofern „Proxyserver verwenden“ aktiviert ist, sollte man den Menüpunkt „Proxyserver für lokale Adressen umgehen“ ebenfalls aktivieren.

Auch Addons für WebBrowser wie "NoScript" können hier Probleme machen. Dabei reicht es z.B. bei "NoScript" nicht in den Einstellungen auf "Skripte allgemein erlauben"zu stellen. In dem Fall muss es deaktiviert werden.

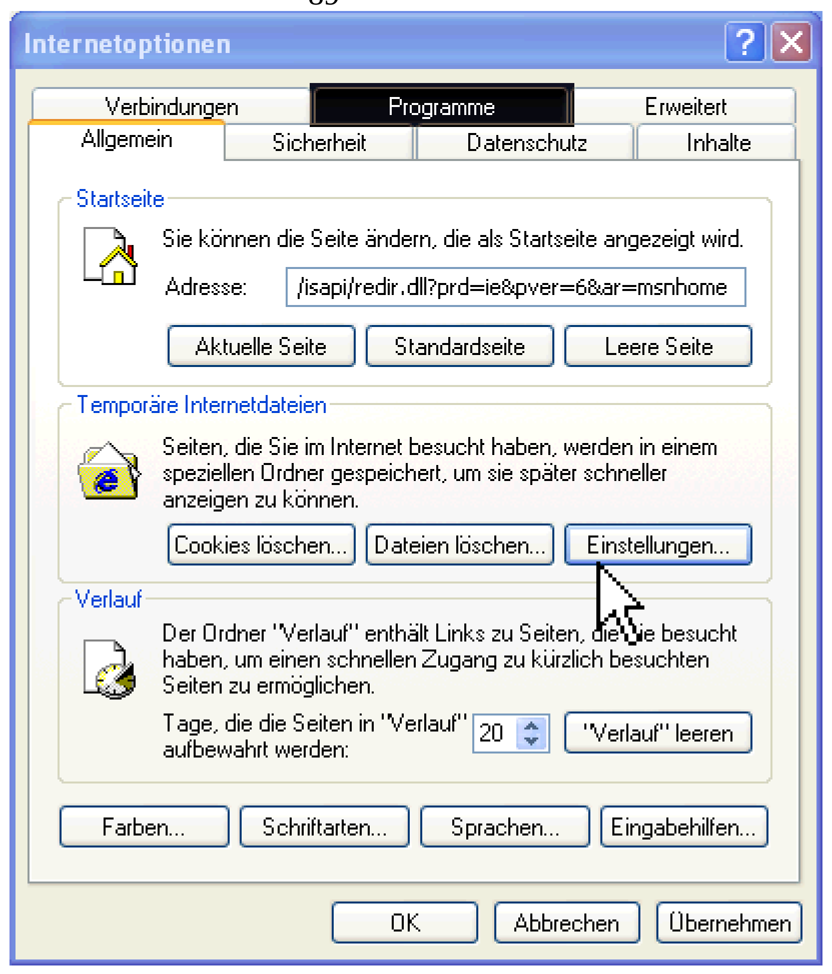
Nach Veränderungen der Einstellungen ist es generell sehr sinnvoll den Cache zu leeren.

### 2.1.6.4Leeren des Browser-Cache

Wenn ein neues SuperX-Applet auf dem Webserver installiert wird, ist es möglich dass die Clients dies nicht sofort mitbekommen. Je nach Java-Version und Betriebssystem unterscheiden Sie sich Wege, den Browser-Cache zu leeren. Unter Windows mit Java 1.4.x wird der Browser-Cache geleert, bei Windows ab Java 1.5.x oder unter Linux wird der Java-Cache geleert. Im Zweifelsfall löschen Sie beide Caches.

Beim Browser-Cache sind die Einstellungen des Browsers maßgeblich. Beim **Internet Explorer** gehen Sie in das Menü "Extras"->"Internetoptionen"

In der Registerkarte "Allgemein" sehen Sie im Abschnitt "Temporäre Internetdateien" den Button "Dateien löschen"; klicken Sie darauf, und löschen Sie alle Inhalte. Danach klicken Sie auf "Einstellungen"…

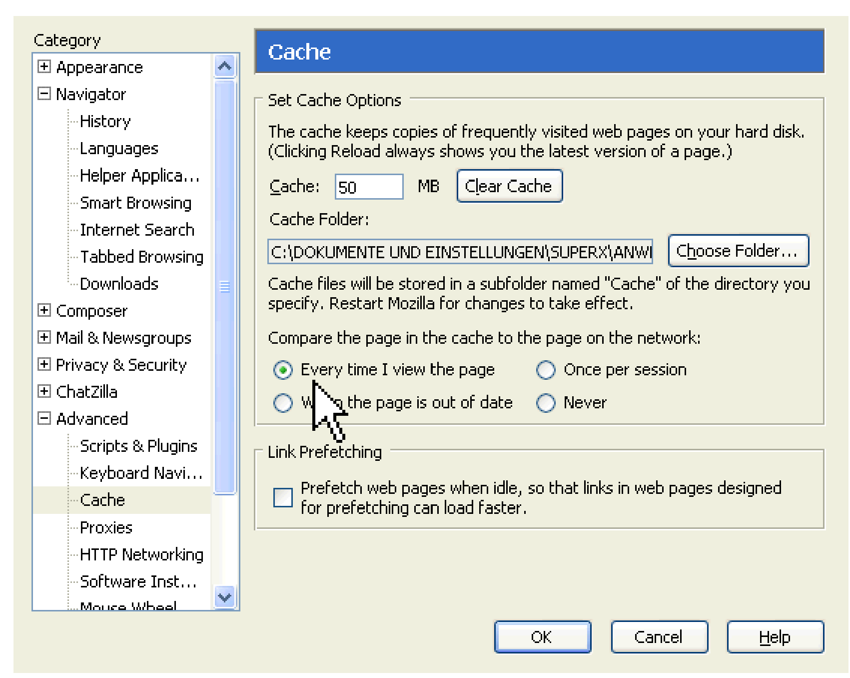



…und markieren Sie den Knopf "Bei jedem Zugriff auf die Seite". Dann drücken Sie "OK".

Starten Sie den Browser dann neu.

Bei **Netscape/Mozilla** befindet sich die Einstellung im Menü "Edit" (deutsch "Bearbeiten") -> "Preferences" (deutsch "Einstellungen").

Hier drücken Sie den Button "Clear Cache" ("Cache leeren") und kreuzen dann unten den Button "Every time I view the page" ("Bei jedem Zugriff auf die Seite") an.




Starten Sie den Browser dann neu.

### 2.1.6.5Leeren des Java - Cache

Bei der Java-Runtime Java 1.5.x unter Windows sowie bei der Java Runtime 1.4.x unter Linux wird ein separater, vom Browser unabhängiger Cache genutzt, der manuell geleert werden muss. Löschen Sie also alle Inhalte in den Pfaden:

Unter Windows:

```
c:\Dokumente und Einstellungen\<<Kennung>>\Anwendungsdaten\
sun\java\deployment\cache\javapi\v.1.0\jar\*
```

Unter Linux:

```
/home/<<Kennung>>/.java/deployment/cache/javapi/v1.0/jar/*
```

Danach starten Sie den Browser neu.

## 2.2Upgrade einer bestehenden SuperX-Installation

Der Update eines bestehenden SuperX ist nicht trivial: Es kursieren verschiedene SuperX-Versionen, und das System ist offen für Änderungen durch den Benutzer. Deshalb müssen die Dateien unterhalb von `$SUPERX_DIR` gesichert werden, und die Datenbank muss vorher exportiert werden. Generell gilt beim Upgrade, dass Sie keinesfalls das normale SuperX-Komplettpaket herunterladen und entpacken sollten, weil dadurch individuelle Konfigurationen überschrieben würden.

Stattdessen sollte Sie immer das passende Upgrade- bzw. "Patch"-Paket herunterladen. Die von Ihnen genutzte Version (zu finden in der Datei $SUPERX_DIR/db/install/VERSION) gibt dazu den besten Anhaltspunkt.

### 2.2.1Patch einspielen

Als erstes müssen Sie sich einen Patch von der Seite http://download.superx-projekt.de/ herunterladen. Dabei ist zu beachten welches System und welche Codierung benötigt wird. Informationen über die Änderungen des Patches finden Sie als Link auf der Downloadseite. In dem Patch selber befindet sich auch noch eine patch_xxxx-xx-xx_readme.txt.

Legen Sie sich für die Patches ein Verzeichnis auf dem SuperX Server an. Am einfachsten wäre z.B. /home/superx/patches. In dieses Verzeichnis kopieren Sie den Patch.

Um Patches in SuperX einzuspielen gibt es das Script $SUPERX_DIR/db/bin/patch_apply.x . Das Script starten Sie direkt aus dem Patchordner, in dem der zu installierende Patch liegt. Gestartet wird es folgendermaßen:

patch_apply.x <<PatchFile>>

Ein Beispiel:

patch_apply.x patch_2011-06-01_superx_iso.zip

Dabei wird in dem Verzeichnis der Patch entpackt und ausgeführt.

### 2.2.2Upgraden des SuperX Kernmoduls auf Version 4.1

Der Upgrade ab dem Kernmodulpaket 4.1 kann auf SuperX Kernmodulen bis zurück zu 3.0 durchgeführt werden. Wenn Sie Kernmodul 3.x installiert hatten und danach immer nur Upgrades gemacht haben, haben Sie noch Tomcat 4 – in diesem Fall müssen Sie einen neuen Tomcat installieren. In diesem Fall empfehlen wir, das Kernmodul- Komplettpaket herunterzuladen.

Wenn Sie Kernmodul 4.0rc1 neu installiert haben, haben Sie bereits Tomcat 5, diese ist mit Kernmodul 4.1 lauffähig, Sie bräuchten also keinen neuen Tomcat und können das Patchpaket nehmen. Bei dem Upgrade mit dem Patchpaket brauchen Sie nur das entsprechende Patchpaket herunterladen, im Superx Verzeichnis entpacken. Dann das Script `$SUPERX_DIR/db/install/upgrade/kern_env_upgrade.x` starten, danach die SQL_ENV neu laden und das Script `$SUPERX_DIR/db/install/upgrade/kern_upgrade.x` starten.

Um sicher zu gehen empfehlen wir mit dem Script `dump_it.x` die Datenbank in einer Datei zu sichern und anschließend von dem gesamten `$SUPERX_DIR` ein Backup anzulegen, bevor Sie mit dem Upgrade beginnen.

Achtung: Bitte installieren Sie vor dem Upgrade SUN/Oracle Java 1.6 oder höher.

#### 2.2.2.1Vorbereitungen für Tomcataktualisierung

1. Kern Paket von der Seite http://download.superx-projekt.de/ herunterladen.
2. Tomcat beenden
3. Auf dem Tomcat-Server das Verzeichnis `$SUPERX_DIR/webserver/tomcat` nach `$SUPERX_DIR/webserver/tomcat_old` kopieren. Dies dient als Backup Verzeichnis und es werden später noch ein paar Dateien davon benötigt.
4. Verzeichnis `$SUPERX_DIR/webserver/tomcat` bis auf den Ordner `webapps` leeren.
5. Verzeichnis `$SUPERX_DIR/webserver/tomcat/webapps/superx/WEB-INF/lib/` leeren.

### 2.2.2.2 Tomcat aktualisieren

1. Kernpaket im `$SUPERX_DIR` entpacken.
2. (Nur bei Mandantenbetrieb) In der `web.xml` bei `de.superx.servlet.SuperXmlAbmeldung` die Parameter `init-param` löschen.
3. Wenn Sie Kernmodul 3.x installierten und bisher immer das Kernmodul geupgradet haben, haben Sie noch Tomcat Version 4. Um dies umzustellen müssen Sie die Connections-Angaben in der `server.xml` in die Datei `webapps/superx/META-INF/context.xml` übertragen (.sam-Datei liegt im gleichen Verzeichnis). Dafür gibt es auch ein Script:
   `sx_transform.x -IN:$SUPERX_DIR/webserver/tomcat_old/conf/server.xml -XSL: $SUPERX_DIR/db/conf/server_xml2context_xml.xsl -OUT:$SUPERX_DIR/webserver/tomcat/webapps/superx/META-INF/content.xml -method:xml`
4. Wenn Sie Kernmodul 4.x installierten, haben Sie bereits Tomcat 5 und damit auch obige Datei context.xml. Diese muss daher nur aus dem alten Tomcat Verzeichnis in das neue übernommen werden.
5. Falls Sie UTF-8 Charset benutzen muss in der Datei `conf/server.xml` bei dem Connector mit dem Port 8080 noch:
   URIEncoding="UTF-8"
   ergänzt werden. Es sieht dann folgendermaßen aus:
   <Connector port="8080" protocol="HTTP/1.1"
   connectionTimeout="20000"
   redirectPort="8443" **URIEncoding="UTF-8"**/>
6. Achten Sie darauf, dass Tomcat die Variable $JRE_HOME benutzt und diese richtig gesetzt ist. Eventuell $JRE_HOME auf $JAVA_HOME/jre setzen.
7. (Nur bei Mandantenbetrieb) $JDBC_CLASSPATH und $XML_CLASSPATH muss für die Mandanten in der SQL_ENV gesetzt werden.

### 2.2.2.3 Datenbank aktualisieren

Bevor Sie den Upgrade ausführen müssen Sie zunächst das Script `$SUPERX_DIR/db/install/upgrade/kern_env_upgrade.x` starten und danach die SQL_ENV neu laden. Nun muss nur noch das Upgradescript `$SUPERX_DIR/db/install/upgrade/kern_upgrade.x` ausgeführt werden.

### 2.2.2.4 Webserver aktualisieren

Wenn Sie den Datenbankserver und Webserver getrennt haben, muss das Kernpacket auch auf dem Webserver entpackt und der ENV UpgradeScript `$SUPERX_DIR/db/install/upgrade/kern_env_upgrade.x` gestartet werden.

Wenn der Apache mit `mod_jk` angebunden ist müssen auch die folgenden Dateien aus dem alten Tomcat übernommen werden:

```
$SUPERX_DIR/webserver/tomcat/conf/superx_mod_jk.conf
$SUPERX_DIR/webserver/tomcat/conf/workers.properties
```

#### 2.2.2.5Falls Joolap instaliert ist

Joolap läuft erst ab der Version 1.2 mit dem Kernmodul 4.1 zusammen. Daher prüfen Sie welche Joolap Version Sie einsetzen und aktualisieren diese gegebenenfalls.

Da die web.xml ersetzt wurde, müssen die Einträge für Joolap wieder eingefügt werden. Eine Anleitung finden Sie dazu in dem Joolap-Admin Handbuch.

Wenn Sie nach dem Kernupgrade Joolap nicht aurufen können und in der Datei `$SUPERX_HOME/webserver/tomcat/logs/catalina.out` steht, dass der ConnectionPool zu der HSQLDB nicht aufgebaut werden kann, da der Treiber fehlt, muss dieser noch in das Tomcat Verzeichnis koppiert werden. Kopieren Sie dann bitte die Datei `$SUPERX_HOME/joolap/lib/joolap.jar` nach `$SUPERX_HOME/webserver/tomcat/lib`.

Danach bitte den Tomcat neu starten.

### 2.2.3Upgrade von Version 3.0 zu 3.5

Der Datenbank-Upgrade ist möglich für vorhandene Kernmodul-Versionen 3.0rc6 bis 3.0final sowie für das Kernmodul 3.1, 3.5 oder 4.x.

- Laden Sie das neueste Patch-Paket `kernmodul_upgrade3x_to_40.tar.gz` von der SuperX-Website herunter und entpacken Sie das Paket auf dem Datenbankserver.
- Gehen Sie in der Shell in das Verzeichnis `$SUPERX_DIR/db/install/upgrade`
- Starten Sie das Script `kernmodul_upgrade_3x_to_35rc2.x`. Es kommt zunächst eine Sicherheitsabfrage, in der auch Ihre aktuelle Version angezeigt wird. Wenn Sie ein Kernmodul 3.0rc5 oder älter einsetzen, müssen Sie zunächst zum Kernmodul 3.0final upgraden.
- Wenn Sie mit **J** bestätigen, startet der Upgrade, die Fehlerausgabe landet in der Datei `upgrade.log`. Wenn kein Fehler auftritt, ist der Datenbank-Upgrade damit abgeschlossen.
  Wenn Sie auch die Webapplikation auf dem Datenbankserver betreiben:
- Stoppen Sie Tomcat, und löschen Sie alle Logdateien in `$SUPERX_DIR/webserver/tomcat/logs`. Starten Sie Tomcat dann neu und prüfen Sie die Ausgabe in der Datei `catalina.out` (unter Unix). Unter Windows heißt die Datei je nach Tomcat-Version `stderr.out` o.ä.

#### 2.2.3.1Komplett-Upgrade: Datenbank und Webapplikation

Der Datenbank-Upgrade ist möglich für vorhandene Kernmodul-Versionen 3.0rc6 bis 3.0final sowie für das Kernmodul 3.5beta oder Kernmodul 3.5rc1.

- Laden Sie das Patch-Paket `kernmodul_upgrade3x_to_35rc2.tar.gz` von der SuperX-Website herunter und entpacken Sie das Paket auf dem Datenbankserver.
- Gehen Sie in der Shell in das Verzeichnis `$SUPERX_DIR/db/install/upgrade`
- Starten Sie das Script `kernmodul_upgrade_3x_to_35rc2.x`. Es kommt zunächst eine Sicherheitsabfrage, in der auch Ihre aktuelle Version angezeigt wird. Wenn Sie ein Kernmodul 3.0rc5 oder älter einsetzen, müssen Sie zunächst zum Kernmodul 3.0final upgraden.
- Wenn Sie mit **J** bestätigen, startet der Upgrade, die Fehlerausgabe landet in der Datei `upgrade.log`. Wenn kein Fehler auftritt, ist der Datenbank-Upgrade damit abgeschlossen.
  Wenn Sie auch die Webapplikation auf dem Datenbankserver betreiben:

- Stoppen Sie Tomcat, und löschen Sie alle Logdateien in `$SUPERX_DIR/webserver/tomcat/logs`. Starten Sie Tomcat dann neu und prüfen Sie die Ausgabe in der Datei `catalina.out` (unter Unix). Unter Windows heißt die Datei je nach Tomcat-Version `stderr.out` o.ä.

### 2.2.3.2 Webapplikation separat Upgraden

Der Webapplikations-Upgrade ist möglich für vorhandene Kernmodul-Versionen ab 3.0final. Die folgenden Schritte sind nur dann notwendig, wenn Sie Datenbank- und Applikationsserver getrennt betreiben.

1. Kopieren und entpacken Sie das Patch-Paket `kernmodul_upgrade3x_to_35rc2.tar.gz` von der SuperX-Website auf dem Applikationsserver.
2. Gehen Sie in der Shell in das Verzeichnis `$SUPERX_DIR/db/install/upgrade`
3. Starten Sie **nicht** das Script `kernmodul_upgrade_3x_to_35rc2.x`, sondern das Script `kernmodul35_webapp_upgrade.x`.
   Es kommt zunächst eine Sicherheitsabfrage, in der auch Ihre aktuelle Version angezeigt wird. Wenn Sie ein Kernmodul 2.1 oder älter einsetzen, müssen Sie zunächst zum Kernmodul 3.0final upgraden.
4. Wenn Sie mit **J** bestätigen, startet der Upgrade, die Fehlerausgabe landet in der Datei `upgrade.log`. Wenn kein Fehler auftritt, ist der Upgrade damit abgeschlossen.
5. Stoppen Sie Tomcat, und löschen Sie alle Logdateien in `$SUPERX_DIR/webserver/tomcat/logs`. Starten Sie Tomcat dann neu und prüfen Sie die Ausgabe in der Datei `catalina.out` (unter Unix). Unter Windows heißt die Datei je nach Tomcat-Version `stderr.out` o.ä.

### 2.2.3.3 Test der Kernmodul-Version 3.5 bei Produktivsystemen

Um das Kernmodul 3.5 zu testen, ohne in das Produktivsystem einzugreifen, sollten Sie eine Testumgebung betreiben. Alternativ können Sie auch einen separaten Applikationsserver aufsetzen, der auf die Produktiv-Datenbank geht. Auf Datenbankebene sind im Kernmodul 3.5 nur wenige Änderungen erfolgt, die zudem abwärtskompatibel sind, d.h. Sie können eine 3.5er Datenbank auch mit der 3.0er-Webapplikation betreiben. Außerdem ist es später kein Problem, auch das Produktivsystem der Webapplikation auf 3.5 umzustellen, weil es dafür ein eigenes Script gibt.

Konkret können Sie also wie folgt vorgehen:

Entscheiden Sie zunächst, ob Sie das Kernmodul 3.5 auf einem neuen Rechner ("Zwei-Server-System") oder auf dem vorhandenen Applikationsserver ("Ein-Server-System") testen wollen. Ersteres ist aufwändiger, aber auch sicherer in Bezug auf mögliche Fehler bei der Konfiguration. Beide Varianten erfordern unterschiedliche Arbeitsschritte:

| Schritt | Ein-Server-System | Zwei-Server-System |
|---|---|---|
| 1 | Kopieren Sie Ihr Produktivsystem in ein neues Verzeichnis (z.B. alle Inhalte unterhalb von `/home/superx` nach `/home/superx/kernmodul3.5`). | Kopieren Sie Ihr Produktivsystem auf einen neuen Rechner (z.B. alle Inhalte unterhalb von `/home/superx` nach `/home/superx` auf dem neuen Rechner). |
| 2 | Sie müssen in der `/home/superx/kernmodul3.5/db/bin/SQL_ENV` die Umgebungsvariable `SUPERX_DIR= /home/superx/kernmodul3.5` setzen. Wenn ein Aufruf der SQL_ENV in der `$HOME/.bashrc` oder `$HOME/.profile` steht, bitte für die Testphase auskommentieren, sonst zeigt die Variable bei jeder neuen Login-Shell auf die alte `SUPERX_DIR` | Sie müssen in der `$SUPERX_DIR/db/bin/SQL_ENV` die Umgebungsvariablen für den Datenbankserver anpassen, also `INFORMIXSERVER` für Informix und `PGHOST` für Postgres.<br>Je nach installiertem Java müssen Sie auch `JAVA_HOME` ändern. |
| 3 | Laden Sie die `SQL_ENV` einmal mit:<br>`. SQL_ENV`<br>und testen Sie, ob die Variable `SUPERX_DIR` auf die neue Installation zeigt:<br>`echo $SUPERX_DIR` | Laden Sie die `SQL_ENV` einmal mit:<br>`. SQL_ENV`<br>und testen Sie, ob Sie sich mit dem DB-Server verbinden können:<br>`psql superx`<br>(bei Postgres) bzw.<br>`dbaccess superx`<br>(bei Informix) |
| 4 | Testen Sie einmal den DB-Zugriff in der Shell mit<br>`DOQUERY "select version from db_version where his_system='kern';"`<br>Als Ergebnis sollte 3.0 kommen | Testen Sie einmal den DB-Zugriff in der Shell mit<br>`DOQUERY "select version from db_version where his_system='kern';"`<br>Als Ergebnis sollte 3.0 kommen |
| 5 | Damit der Produktiv-Tomcat und der Test-Tomcat sich nicht in die Quere kommen, müssen Sie in der Datei `/home/superx/kernmodul3.5/webserver/tomcat/conf/server.xml` andere Ports angeben (z.B. von 8005,8009,8080 zu 9005,9009,9080). | Für den neuen Tomcat müssen Sie in den Dateien `$SUPERX_DIR/webserver/tomcat/conf/server.xml` und `$SUPERX_DIR/webserver/tomcat/webapps/superx/WEB-INF/db.properties` den Hostnamen für den JDBC-Zugriff ändern, häufig steht hier "localhost", dies müssen Sie ändern zum Rechnernahmen des DB-Servers. |
| 6 | Starten Sie den Test-Tomcat und prüfen Sie, ob die Anmeldung im XML-Frontend klappt. Die Fehlerausgabe steht in `/home/superx/kernmodul3.5/webserver/tomcat/logs/catalina.out` | Starten Sie den Test-Tomcat und prüfen Sie, ob die Anmeldung im XML-Frontend klappt. Die Fehlerausgabe steht in `$SUPERX_DIR/webserver/tomcat/logs/catalina.out` |
| 7 | Laden Sie das Kernmodul 3.0-Upgrade-Paket, und entpacken Sie es in `/home/superx/kernmodul3.5` | Laden Sie das Kernmodul 3.0- Upgrade -Paket, und entpacken Sie es in `$SUPERX_DIR` |
| 8 | Unterhalb von `/home/superx/kernmodul3.5` gehen Sie nun vor wie beim Komplett-Upgrade beschrieben (Punkt 1, das Herunterladen und Entpacken, haben Sie schon erledigt). | Innerhalb der `$SUPERX_DIR` gehen Sie nun vor wie beim Komplett-Upgrade beschrieben (Punkt 1, das Herunterladen und Entpacken, haben Sie schon erledigt). |
| 9 | Testen Sie wieder den DB-Zugriff in der Shell mit<br>`DOQUERY "select version from db_version where his_system='kern';"`<br>Als Ergebnis sollte 3.5* kommen | |

| | | |
|---|---|---|
| 10 | Da nun zwei Tomcats gleichzeitig auf eine Datenbank gehen, müssen Sie prüfen, ob die Anzahl der gleichzeitig möglichen Datenbankverbindungen nicht von den Größen der Connection Pools der DBFORMS und des SuperX-Servlets im Applikationsserver überschritten wird. Da es sich bei dem Test-Tomcat nur um ein Testsystem handelt, können Sie dort den Connection Pool auf eine geringe Zahl (z.B. 5 für DBFORMS und SuperX-Servlet) reduzieren. | |
| 11 | Starten Sie dann den Test-Tomcat in `/home/superx/kernmodul3.5/webserver/tomcat/bin` mit `startup.sh` | Starten Sie dann den Test-Tomcat in `$SUPERX_DIR/webserver/tomcat/bin` mit `startup.sh` |

Damit haben Sie einen Test-Tomcat mit dem Ajax-Client, der auf Ihre vorhandene Datenbank geht und damit alle Abfragen und Inhalte anbietet.Gleichzeitig kann der Produktiv-Tomcat weiter arbeiten. Sie können also in Ruhe testen!

#### 2.2.3.4Upgrade bei mehreren Mandanten

Wenn Sie einen mandantenfähige SuperX-Installation upgraden, müssen Sie auf einen Schlag alle Mandanten aktualisieren, es ist nicht möglich einzelne Mandanten mit der älteren Version arbeiten zu lassen.

Der **Datenbank-Upgrade** bei mandantenfähigen Installationen verhält sich genauso wie der Upgrade einer Einzelplatz-Installation - mit einer Ausnahme: Die Umgebung für den Mandanten wird nicht in der Datei `$SUPERX_DIR/db/bin/SQL_ENV` gespeichert, sondern in einer speziellen Mandaten-Datei, z.B. `$SUPERX_DIR/db/bin/SQL_ENV_PHHD`. Da das Upgrade-Script diesen Dateinamen nicht kennt, muss eine Änderung manuell vollzogen werden: In der Datei müssen alle Nennungen von der Datei "`superx3.0.jar`" geändert werden nach "`superx3.5.jar`" (z.B. in Variable `JDBC_CLASSPATH`).

Der **Upgrade der Webapplikation** ist entspricht dem Vorgehen wie oben gezeigt, mit einer Ausnahme: Sie müssen wie gehabt über das Script

```
$SUPERX_DIR/webserver/tomcat/webapps/superx/upgradeMandantendir <<MANDANTID>>
```

jeden einzelnen Mandanten aktualisieren. Im Kernmodul 3.5 wurde korrigiert, dass die der Datei `$SUPERX_DIR/webserver/tomcat/webapps/superx/<<MANDANTID>>/xml/anmeldung.htm` das versteckte Feld "mandantid" nicht mehr überschrieben wird, Sie können also sofort loslegen.

### 2.2.4Kurzanleitung zum Upgrade von Version 2.1 nach 3.0

Die folgende Kurzanleitung zeigt die wesentlichen Schritte für den Upgrade von 2.1 nach 3.0.

1.Stoppen Sie tomcat über

```
$SUPERX_DIR/webserver/tomcat/bin/shutdown.sh
```

2.Sichern Sie den alten Tomcat

```
mv $SUPERX_DIR/webserver/tomcat $SUPERX_DIR/webserver/tomcat_alt
```

3.Sichern Sie das gesamte Verzeichnis `$SUPERX_DIR`

1.Entpacken Sie das neue Kernmodul unter `$SUPERX_DIR`

```
cd $SUPERX_DIR
tar -xzvf kernmodul3.0.tar.gz
```

1. Wenn Sie Datenbank- und Webserver auf unterschiedlichen Rechnern betreiben, müssen Sie das Kernmodul auf beiden Rechnern entpacken.

Die alten Scripte werden dadurch überschrieben, nicht aber die vorhandenen Properties-Dateien (unsere properties-Dateien sowie auch das Access-Frontend haben immer die Endung .sam für sample).

#### 2.2.4.1 Der Datenbankupgrade

1. Gehen Sie in das Verzeichnis $SUPERX_DIR/db/bin
2. Kopieren Sie die Datei SQL_ENV nach SQL_ENV.alt
3. Kopieren Sie SQL_ENV.sam nach SQL_ENV
4. Übertragen Sie relevante Parameter aus SQL_ENV.alt nach SQL_ENV, z.B. SX_CLIENT, JAVA_HOME, DATABASE, MAILPROG.
5. Sourcen Sie die neue SQL_ENV mit
   . SQL_ENV
6. Gehen Sie in das Verzeichnis $SUPERX_DIR/db/install
7. Sichern Sie die Datenbank mit dump_it.x
8. Gehen Sie in das Verzeichnis upgrade
9. Starten Sie das Script kernmodul_upgrade21_to_30.x, und prüfen Sie auf Fehler in der Logdatei upgrade.log
10. Entfernen Sie aus dem Themenbaum den "alten" Ast Administration. Sie erkennen diesen daran, dass im Klammern (alt) dehinter steht. Wenn Sie eigene Masken dort erzeugt haben, müssen Sie diese in den neuen Ast verschieben.
11. Richten Sie bei Bedarf das neue Access-Frontend ein (Kopieren von $SUPERX_DIR/db/superx_frontend_sam.mdb nach superx_frontend.mdb).

#### 2.2.4.2 Upgrade des Webservers

1. Kopieren Sie die Dateien superx.properties und db.properties
   cp $SUPERX_DIR/webserver/tomcat_alt/webapps/superx/WEB-INF/db.properties $SUPERX_DIR/webserver/tomcat/webapps/superx/WEB-INF/db.properties
   cp $SUPERX_DIR/webserver/tomcat_alt/webapps/superx/applet/superx.properties
   $SUPERX_DIR/webserver/tomcat/webapps/superx/applet/superx.properties
2. Nur bei Informix: Stellen Sie sicher, dass sich eine aktuelle ifxjdbc.jar unter
   $SUPERX_DIR/webserver/tomcat/webapps/superx/WEB-INF/lib
   befindet. Achtung: Es ist eine neuere Version als für SuperX 2.1 nötig!

1. Benennen Sie die Datei
   $SUPERX_DIR/webserver/tomcat/webapps/superx/WEB-INF/kern_dbforms_config_<<DB-Kürzel>>.xml um nach
   $SUPERX_DIR/webserver/tomcat/webapps/superx/WEB-INF/dbforms_config.xml
2. Tragen Sie in der Datei $SUPERX_DIR/webserver/tomcat/conf/server.xml im Kontext superx die JDBC-Parameter nach (url, username, password, maxIdle).
3. Starten Sie Tomcat neu, und testen Sie die Anmeldung im XML-Frontend und im Applet
4. Mit der neuen Webanwendung ist auch eine neue web.xml im Verzeichnis
   $SUPERX_DIR/webserver/tomcat/webapps/superx/WEB-INF/web.xml installiert worden
   Wenn Sie die maxRows geändert haben, dann sollten Sie die Änderungen manuell von der tomcat-Datei nachtragen.

Der Upgrade von SuperX-Karlsruhe (Win32-Client und Informix-Datenbank) gestaltet sich anders und wird im unten beschrieben.

### 2.2.5Kurzanleitung zum Upgrade von Version 2.0 nach 2.1

Die folgende Kurzanleitung zeigt die wesentlichen Schritte für den Upgrade von 2.0 nach 2.1.

1.Stoppen Sie tomcat über

`$SUPERX_DIR/webserver/tomcat/bin/shutdown.sh`

2.Sichern Sie den alten Tomcat

`mv $SUPERX_DIR/webserver/tomcat $SUPERX_DIR/webserver/tomcat3`

3.Sichern Sie das gesamte Verzeichnis `$SUPERX_DIR`

1.Entpacken Sie das neue Kernmodul unter `$SUPERX_DIR`

```
cd $SUPERX_DIR
tar -xzvf kernmodul2.1.tar.gz
```

Wenn Sie Datenbank- und Webserver auf unterschiedlichen Rechnern betreiben, müssen Sie das Kernmodul auf beiden Rechnern entpacken.

Die alten Scripte werden dadurch überschrieben, nicht aber die vorhandenen Properties-Dateien (unsere properties-Dateien sowie auch das Access-Frontend haben immer die Endung `.sam` für sample).

#### 2.2.5.1Der Datenbankupgrade

1.Gehen Sie in das Verzeichnis `$SUPERX_DIR/db/bin`
2.Kopieren Sie die Datei `SQL_ENV` nach `SQL_ENV.alt`
3.Kopieren Sie `SQL_ENV.sam` nach `SQL_ENV`
4.Übertragen Sie relevante Parameter aus `SQL_ENV.alt` nach `SQL_ENV`, z.B. `SX_CLIENT`, `JAVA_HOME` etc.
5.Sourcen Sie die neue SQL_ENV mit
```
. SQL_ENV
```
6.Gehen Sie in das Verzeichnis `$SUPERX_DIR/db/install`
7.Sichern Sie die Datenbank mit `dump_it.x` (Bei Informix müssen Sie vorher alle Datenbankverbindungen, d.h. auch Tomcat, beenden).
8.Gehen Sie in das Verzeichnis `upgrade`
9.Starten Sie das Script `kernmodul_upgrade20_to_21.x`, und prüfen Sie auf Fehler in der Logdatei `upgrade.log`[19]
10.Ggf. müssen Sie das Script `macro_ids.sql` bzw. `macro_pg.sql` von Hand starten, wenn die Tabelle `macro_masken_bez` nicht das Feld `nummer` enthält.
11.Entfernen Sie aus dem Themenbaum den "alten" Ast Administration. Sie erkennen diesen daran, dass darunter weniger Masken sind als unter dem neuen Ast.
12.Richten Sie bei Bedarf das neue Access-Frontend ein (Kopieren von `$SUPERX_DIR/db/superx_frontend_sam.mdb` nach `superx_frontend.mdb`).

---

[19] Das Schwierigste ist der Update der Datenbank. Dies läßt sich zwar automatisch durchführen, aber wegen der heterogenen Datenbankstrukturen kann es zu Fehlern kommen. Am sichersten ist es, die Schritte im Update-Script `$SUPERX_DIR/db/install/update/kernmodul-upgrade20_to_21_ids.sql` (für Postgres `upgrade20_to_21_pg.sql`) einzeln durchzuführen.

#### 2.2.5.2Upgrade des Webservers

1.Kopieren Sie die Dateien superx.properties und db.properties

```
cp $SUPERX_DIR/webserver/tomcat3/webapps/superx/WEB-INF/db.properties
$SUPERX_DIR/webserver/tomcat/webapps/superx/WEB-INF/db.properties
cp $SUPERX_DIR/webserver/tomcat3/webapps/superx/applet/superx.properties $SUPERX_DIR/webserver/tom-
cat/webapps/superx/applet/superx.properties
```

2.Nur bei Informix: Kopieren Sie die Datei
$SUPERX_DIR/webserver/tomcat3/lib/ifxjdbc.jar
nach
$SUPERX_DIR/webserver/tomcat/webapps/superx/WEB-INF/lib

3.Starten Sie Tomcat neu, und testen Sie die Anmeldung

4.Mit Tomcat 4 ist auch eine neue web.xml im Verzeichnis
$SUPERX_DIR/webserver/tomcat/webapps/superx/WEB-INF/web.xml installiert worden
Wenn Sie die Logging-Parameter oder die maxRows geändert haben, dann sollten Sie die Änderungen manuell von der tomcat3-Datei nachtragen.

Das gleiche Verfahren gilt nicht nur für das Kernmodul, sondern auch für die einzelnen Module. Die Update-Scripte liegen jeweils in

```
$SUPERX_DIR/db/module/<<Modulname>>/<<Modulname>>_modul_upgrade<<Versionsnr>>.x
```

### 2.2.6Upgrade von SuperX Karlsruhe auf SuperX V2.0

Das Kernmodul der SuperX-Version 2 ist abwärtskompatibel mit der SuperX-Anwendung aus Karlsruhe, wenn ein paar Änderungen an der Datenbank vorgenommen werden. Bevor das neue Servlet und das neue Applet eingesetzt werden, sollten die notwendigen Tabellen erzeugt worden sein. Der Upgrade ist leider nicht automatisch machbar, weil frühere SuperX-Installationen sehr unterschiedlich sind. Es ist daher möglich, dass einige Scripte mit Fehler abbrechen.

#### 2.2.6.1Erzeugen der Tabellen

Sie erzeugen bzw. erweitern Tabellen und Prozeduren in der SuperX-Datenbank. Diese Änderungen sollten nicht im laufenden Betrieb von SuperX vorgenommen werden, da die neuen Tabellen, z.B. bis auf userinfo und user_institutionen vom alten SuperX benutzt werden. Außerdem sollten Sie die Regel beherzigen, zunächst den bestehenden Stand der Datenbank mit dbexport zu sichern.

Bisherige SuperX-Benutzer können mit der vorliegenden SuperX-Version ihre Masken weiterverwenden. Dazu müssen die Masken und Sachgebiete in den Themenbaum übernommen werden. Dazu liegt ein Script im Verzeichnis

```
$SUPERX_DIR/db/install/update_von_superx_karlsruhe/einmal_starten/
themenbaum_import_superxalt.sql
```

Außerdem können bisherige SuperX-Benutzer mit der vorliegenden SuperX-Version ihr Organigramm weiterverwenden. Dazu müssen neben den Masken die Tabellen `Bereiche`, `Einrichtungen`, `Institutionen` und `Abteilungen` in das Organigramm übernommen werden. Dazu liegt ein Script im Verzeichnis

```
$SUPERX_DIR/db/install/update_von_superx_karlsruhe/einmal_starten/
organigramm_import_superxalt.sql
```

Die Tabelle `hochschulinfo` darf nur mit einem Datensatz, dem Namen und der Nummer der Hochschule gefüllt sein. Das Script fügt auch die Lehreinheiten unterhalb der Hochschule unter dem Knoten "Lehreinheiten" hinzu; dadurch können Sie mit dem Admin-Tool die Lehreinheiten einfacher in das Organigramm übernehmen. Es bleibt dabei Ihnen überlassen, ob Sie die Lehreinheiten in die Institutionsstruktur einfügen oder nicht.

Die Tabelle `userinfo` wird um zwei deskriptive Felder erweitert: "name" für den Benutzernamen, und "info" für die Beschreibung der Person. Außerdem wird das Feld `passwd_sha` hinzugefügt, und als Default das Passwort "anfang12" gesetzt.

```
$SUPERX_DIR/db/install/update_von_superx_karlsruhe/einmal_starten/
userinfo_import_superxalt.sql
```

Die Tabelle `user_institution` im "alten" SuperX wurde erweitert um die Felder gueltig_seit, gueltig_bis und lehre; das Script

```
$SUPERX_DIR/db/install/update_von_superx_karlsruhe/einmal_starten/
user_institution_import_superxalt.sql
```

konvertiert die Tabelle in das neue Format.

Die Tabelle `felderinfo` im "alten" SuperX wurde geändert: das Feld default lautet nun defaultwert; dadurch ist SuperX mit anderen Datenbanken kompatibel, z.B. PostgreSQL; das Script

```
$SUPERX_DIR/db/install/update_von_superx_karlsruhe/einmal_starten/
felderinfo_import_superxalt.sql
```

konvertiert die Tabelle in das neue Format. Achtung: Der darin enthaltene Passus, das in der Tabelle felderinfo auch das Feld default gelöscht wird, macht SuperX inkompatibel zum Win32-Client aus Karlsruhe. Andererseits funtionieren ohne droppen des Feldes die Scripte sx_select_mask und sx_insert_mask nur noch mit eigenen Abfragen, nicht mit Download-Versionen anderer Hochschulen.

Der Tabelle `maskeninfo` im "alten" SuperX wurde das Feld hinweis hinzugefügt. Die ermgöglicht es, zu Ergebnistabellen Legenden zu liefern, die im Kopf der Tabelle angezeigt werden; das Script

```
$SUPERX_DIR/db/install/update_von_superx_karlsruhe/einmal_starten/
maskeninfo_import_superxalt.sql
```

konvertiert die Tabelle in das neue Format.

| **Die Tabellenübernahme - Kurzanleitung:** |
|---|
| Starten Sie im DBACCESS nacheinander die folgenden Scripte im Verzeichnis<br>`$SUPERX_DIR/db/install/update_von_superx_karlsruhe/einmal_starten/`<br>(Sicherheitsabfragen können Sie ggf. mit Ja beantworten):<br>`themenbaum_import_superxalt.sql`<br>`organigramm_import_superxalt.sql`<br>`userinfo_import_superxalt.sql`<br>`user_institution_import_superxalt.sql`<br>`felderinfo_import_superxalt.sql` |

```
maskeninfo_import_superxalt.sql
macro.sql
```

Sie erzeugen so die Tabellen `Themenbaum`, `Organigramm` und erweitern die anderen Tabellen.

#### 2.2.6.2 Erzeugen der Prozeduren

Der Themenbaum benötigt keinerlei Prozeduren, aber das Organigramm benötigt die Prozeduren

```
sp_user_orga.sql
sp_user_orga_child.sql
```

Die Prozeduren liegen im Verzeichnis

`$SUPERX_DIR/db/install/update_von_superx_karlsruhe/prozeduren/`

Der Dateiname beginnt wie in SuperX üblich mit proc_. Erzeugen Sie diese Prozeduren im DBACCESS.

#### 2.2.6.3 Upgrade des Servlets und Applets

Das SuperX-Servlet ist nicht allein updatebar, bei einem Upgrade muss das gesamte Verzeichnis `$SUPERX_DIR/webserver` gesichert, gelöscht, und dann neu entpackt werden, wie in der Installationsanleitung des Webservers beschrieben. Danach müssen die Dateien `db.properties` sowie `superx.properties` vom "alten" Webserver zum neuen Webserver kopiert werden.

#### 2.2.6.4 Ändern der Masken

Die alten SuperX-Masken sind problemlos im neuen SuperX lauffähig. Falls Sie schon den Karlsruher SuperX-Client (WIN32-Anwendung) in Betrieb haben, müssen Sie beachten, dass sich der alte Client teilweise anders als der Java-Client verhält. Umgekehrt ist das neue SuperX-Kernmodul mit Version 2.0 nicht mehr kompatibel mit dem alten Client. Beide Clients können also lediglich für Spezialabfragen parallel betrieben werden (z.B. aus 'historischen' Gründen). Beim Anpassen der alten SuperX-Abfragen für SuperX Version 2.0 müssen die folgenden Punkte beachtet werden.

#### 2.2.6.5 Masken-Anpassung für den Java-Client

Die Masken werden je nach Client auf unterschiedliche Arten aufgebaut. Für die Gestaltung der Masken sind die Tabellen *maskeninfo* und *felderinfo* verantwortlich.

##### 2.2.6.5.1 Felderinfo

- Das Feld `default` wurde umbenannt in `defaultwert`

- Bei Mehrfachauswahl-Feldern muss als Datentyp immer char gewählt werden
- Bei "char(xx)"-Deklarationen zeigt der Java-Client keine Textfelder an. Man muss die Typdeklaration ändern auf "char"
- Attribut **Art**: 6 fällt als Zahl weg. Statt 6 wählt man 5 (nur Anzeige) - 0 ist die einzige Art, bei der nur ein Feld angegeben wird, ansonsten immer 2; zur Not select semester, semester
- Bei Auswahl Köpfe oder Fälle funktioniert nur der Datentyp sql, nicht char oder integer, bei beiden Clients
- Bei Eingabe eines Datums muss zwingend der Datentyp `date` gesetzt werden. Leider muss man dies für den alten client im `select_stmt` der Tabelle maskeninfo mit date("<<feld>>") abfangen

#### 2.2.6.5.2Maskeninfo

- Der Tabelle wurde das Feld `hinweis` hinzugefügt, dass es ermöglicht, Ergebnistabellen zu dokumentieren.
- Bei Ergebnistabellen im neuen Client wird nicht mehr automatisch gerundet round(<<Ausdruck>>,2)
- Bei der Vorgabe des Tabellenlayouts (XIL-Proplist) sind unbedingt Leerzeichen zwischen den Attributen erforderlich. Im Zweifelsfall muss man die Beschreibung von einer funktionierenden Maske kopieren.

Auch der Aufruf des Organigramms in den Masken muss geändert werden. Die folgende Zeile im Feld select_stmt der Tabelle `maskeninfo`

```
execute procedure sp_ch110_institut (/* <<Institution>> +1 */ -1, <<UserID>>);
```

muss ersetzt werden durch

```
execute procedure sp_user_orga_child (<<UserID>>, <<Organigramm-Stand>>, 0,
 /* <<Institution>> +1 */ -1,<<erlaubt>>);
```

## 2.3Datenschutz

Allgemeine Hinweise siehe Datenschutzdokumentation.

### 2.3.1Checkliste Sicherheitsmaßnahmen SuperX

#### 2.3.1.1Keine Verwendung von Standardkennungen

Verwenden Sie nach Möglichkeit nicht die Standardkennungen (superx, admin und testuser), die bei Auslieferung im SuperX-Kernmodul enthalten sind. Richten Sie eine bzw. mehrere neue Administrator-Kennungen an und arbeiten mit diesen. Die Standardkennungen superx, admin, testuser sollten aus der SuperX-Tabelle `userinfo` gelöscht werden.

#### 2.3.1.2Applet deaktivieren

Um das logging im Applet abzuschalten, setzen Sie in `$WEBAPP/applet/superx.properties` „logToKonsole“ auf „none“.

Falls Sie das Applet nicht benötigen löschen Sie das Verzeichnis
`$WEBAPP/applet`

Außerdem wird die Sicherheit erhöht, wenn ein Zugriff auf das nur vom Applet benutzte Servlet SuperXDBServlet unterbunden wird.

Bearbeiten Sie dazu Ihre Datei `$WEBAPP/WEB-INF/web.xml`

**Kommentieren Sie das Servlet aus mit den Zeichen <!-- und -->**

```
<!-- ************************ SUPERXDBSERVLET
********************************** -->
<!--
<servlet>
      <servlet-name>SuperXDBServlet</servlet-name>
      <servlet-class>de.superx.servlet.SuperXDBServlet</serv-
let-class>

<init-param>
...
</init-param>
</servlet>
-->
```

**Ergänzen Sie ein Eintrag**

```
<servlet>
<servlet-name>de.superx.servlet.SuperXDBServlet</servlet-
name>
<servlet-class>xxx</servlet-class>
</servlet>
```

Starten Sie danach Tomcat neu.

#### 2.3.1.3Public-Private-Key-Kontrolle von Applet-Befehlen

Das Applet genügt von seiner Anlage her nicht mehr den modernen Sicherheitsanforderungen und wird mit dem Kernmodul 3.5 durch das XML-Frontend ersetzt. Wenn Sie das Applet dennoch einsetzen wollen: Zur Erhöhung der Sicherheit ist es möglich, eine DSA-public/private-Key-Kontrolle zu installieren. Dabei wird jeder Befehl, der vom Applet ans Servlet geschickt wird, mit dem einen Key signiert. Im Servlet wird mit Hilfe des anderen, nur dort bekannten Keys kontrolliert, ob der ankommende Befehl eine gültige Signatur aufweist.

Im Applet können Sie den Info-Button anklicken, in der erscheinenden Infobox wird angegeben, ob public/private key Kontrolle aktiv ist oder nicht.

#### 2.3.1.4Datenbankverbindung über einen eingeschränkten Datenbank-User

Zur Erhöhung der Sicherheit ist es möglich, dass die Datenbankverbindung von Tomcat zur Datenbank mit einem eingeschränkten User durchgeführt wird. Dies wird von ZENDAS (Zentrale Datenschutzstelle der baden-württembergischen Universitäten) für den Produktivbetrieb nachdrückliche empfohlen.

Richten Sie dazu einen entsprechenden eingeschränkten User in Ihrer Datenbank ein und geben Sie diesen beim Propadmin bei *eingeschränkter User* an. Der erste im Propadmin auszufüllende User muss weiterhin umfassende Rechte auf alle Tabellen haben, weil er auch bei Komponenteninstallationen/-updates verwendet wird. Das Minimum, was der eingeschränkte User haben muss sind select Rechte auf alle Tabellen, insert-Rechte auf die Tabelle `user_pw` und `protokoll` (sowie bei Postgres auf die zugehörige Sequence `protokoll_protokoll_id_seq`) und update-Rechte auf `userinfo`.

Sobald Sie Ihre `db.properties` mit dem Propadmin bearbeitet haben, können Sie praktisch die Minimal nötigen Rechte vergeben, in dem Sie einmal das Skript

`sx_restrictedconnmanager.x false` aufrufen. Rufen Sie dieses Skript erneut auf nach Komponentenneuinstallationen oder Upgradepatches.

Nach einem Tomcat-Neustart findet sich in der `catalina.out` nach "Aufbau von Datenbank-Connection-Pool (..) .. OK" ein Hinweis:

```
eingeschränkter Datenbankuser für Verbindung: true|false
```

**Wenn Sie Funktionen wie User/Gruppe/Maske einrichten/löschen etc. im XML-Frontend benutzen wollen, müssen zusätzliche Kernmodultabellen freigeschaltet werden:**

```
protokoll
userinfo
groupinfo
user_institution
user_sachgeb_bez
user_masken_bez
group_sachgeb_bez
group_masken_bez
user_group_bez
user_pw
user_sichten
user_sichtarten
group-sichten
group_sichtarten
felderinfo
maskeninfo
maske_system_bez
masken_felder_bez
sachgeb_maske_bez
organigramm
themenbaum
```

Am einfachsten können Sie dies erledigen, indem Sie das Skript

`sx_restrictedconnmanager.x true` aufrufen.

Exkurs:

Wenn Sie die höchste Sicherheit wollen, aber der Zuständige für die Userverwaltung trotzdem das XML-Frontend benutzen können soll, könnten Sie folgendermaßen vorgehen:

- Richten Sie für den regulären Betrieb einen eingeschränkten User mit minimalen Rechten ein, wie oben beschrieben und deaktivieren Sie alle Datenbankformulare, indem Sie nach jedem Komponentenupgrade das Verzeichnis $WEBAPP/edit leeren.
- Erzeugen Sie einen weiteren eingeschränkten Datenbankuser, der zusätzlich die Kernkomponententabellen bearbeiten darf.
- Richten Sie einen zweiten Tomcat ein, der mit diesem zweiten eingeschränkten Datenbankuser arbeitet.
- Sorgen Sie (z.B. per Firewall) dafür, dass nur der für die Userverwaltung zuständige Mitarbeit Zugriff auf den zweiten Tomcat hat.

#### 2.3.1.5Entfernen von temporären Dateien

Entfernen Sie temporäre Dateien, die sich auf dem Webserver befinden (z.B. mit Endung ~ oder #Untitled#). In $SUPERX_DIR/db/bin steht das Skript remove_tmp.x zur Verfügung. Es entfernt automatisch alle Dateien mit den Endungen ~, tmp und bak sowie #Untitled#-Dateien aus dem aktuellen Verzeichnis und dessen Unterverzeichnissen. Optional kann auch ein Pfad angegeben werden, in dem die Dateien gelöscht werden sollen, z.B.: remove_tmp.x $WEBAPP

## 2.4Das Clientpaket

Wenn Sie nicht das gesamte Kernmodul inkl. Tomcat benötigen, sondern nur ein kleines Paket, um regelmäßige Administrationsaufgaben zu erledigen, haben wir ein Clientpaket "geschnürt", das die wichtigsten Werkzeuge zur mit dem DWH beinhaltet. Insbesondere Windows-Anwender können dieses Paket benutzen, um mit dem DWH zu arbeiten, z.B. Masken entwickeln, Tabellen entladen etc. Dazu enthält das Paket ein paar Werkzeuge.

Das Client-Paket wird außerdem für das Entladen aus HIS-Systemen unter Windows genutzt.

### 2.4.1Installation

Laden Sie das Paket client<<Versionsnummer>>_<<Codierung>>.zip und speichern Sie es lokal auf der Festplatte.

#### 2.4.1.1Einrichten der Umgebung

Entpacken Sie dieses Archiv in einem separaten Verzeichnis, z.B. c:\Programme\edustore_client, und benennen Sie die Dateien um; unter Windows:

bin\client_env_sam.bat nach bin\client_env.bat

bzw. unter Unix:

bin/client_env.x.sam nach bin/client_env.x

Passen Sie die Parameter in der Datei an, und sorgen Sie bei Bedarf dafür, daß die Datei beim Aufruf der Shell ausgeführt wird (unter DOS autoexec.bat, unter Linux / bash die ~/.bashrc).

Folgende Parameter müssen Sie wahrscheinlich anpassen:

JAVA_HOME (Der Pfad zur JRE)

CLIENT_DIR (das Verzeichnis, in dem Sie den Client entpackt haben)

Folgende Parameter sind wichtig, aber meist korrekt vorbelegt:

JDBC_CLASSPATH (der Pfad zu Ihrem jdbc-Treiber)

DB_PROPERTIES (der Pfad zu den Datenbankparametern)

Wenn die Datei fertig eingerichtet ist, wird sie wie folgt in die aktuelle Shell geladen:

Unter DOS:

```
client_env.bat
```

Unter Unix:

```
. client_env.x
```

#### 2.4.1.2Einrichtung einer Datenbankverbindung

Mit dem Clientpaket können Sie u.a. auf die DWH-Datenbank zugreifen. Um dies zu tun, werden die Verbdingsdaten in einer properties-Datei gespeichert, die standardmäßig im Verzeichnis `conf` gespeichert wird. Starten Sie zunächst den `propadmin` mit

`propadmin.bat` (DOS)

bzw.

`propadmin.x` (Linux)

Dort geben Sie die Parameter für den DB-Zugriff ein. Das Passwort wird verschlüsselt

gespeichert. Danach sind die Kommandozeilen-Werkzeuge verfügbar. Unter Unix sind alle dort genannten Scripte nutzbar, unter DOS nur eine Auswahl (erkennbar daran, dass es eine Datei mit der Endung ".bat" im bin-Verzeichnis gibt).

### 2.4.2Weitere Werkzeuge

Im Clientpaket befinden im Ordner tools die Anwendungen Jedit sowie die sqlWorkbench sowie das Access-Frontend. Diese Tools dienen zur Abfragenentwicklung. Details dazu finden Sie im Entwicklerhandbuch.

### 2.4.3Download von Berichtsausgaben

Sie können mit dem Clientpaket auch Berichtsausgaben automatisch vom Server herunterladen. Dazu müssen Sie zunächst eine Kennung einrichten, die die entsprechenden Rechte besitzt. Wenn Sie die Kennung in HISinOne pflegen, müssen Sie sich einmalig in den Grunddaten- und Basisberichten anmelden, und danach muss der Administrator dieser Kennung ein "echtes" Passwort zuweisen.

Wenn dies geschehen ist, können Sie sich zunächst im Browser einen Link zusammenbasteln, der den Bericht ohne Login-Dialog anzeigt. Die URL dafür lautet:

```
http(s)://Servername:Port/superx/servlet/SuperXmlTabelle?
tid=<<Maskennummer>>&kennung=<<Kennung>>&passwort=<<Passwort>>&<<ggf. weite-
re Parameter>>
```

wobei die Maskennummer für die eindeutige Nummer der Maske steht. Sie erfahren die Maskennummer, indem Sie die jeweilige Modulbeschreibung konsultieren, oder indem Sie einfach die Maus über den Link halten, dann wird die Nummer im Browser unten in der Statusleiste angezeigt.

Neben der Maskennummer muss die Kennung und das Passwort übergeben werden, sowie je nach Maske weitere Felder.

Einfaches Beispiel: Das Prüfprotokoll in der Komponente Stellen, Personal:

```
http://localhost:8080/superx/servlet/SuperXmlTabelle?tid=19220&kennung=su-
perx&passwort=anfang12
```

Wenn diese Link im Browser funktioniert, können Sie die Datei wie folgt im Excel-Format herunterladen:

DOS (Achtung: die Zeichen "=" und "&" müssen mit Caret-Zeichen ("^") vorangestellt werden, außerdem muss die URL in Anführungszeichen gesetzt werden, sonst klappt die Parameterübergabe in DOS nicht):

```
wget.bat "http://solomon:8080/superx/servlet/SuperXmlTabelle^?tid^=19220^&kennung^=superx^&passwort^=anfang12" pruefprotokoll.xls
```

Unix:

```
wget.x 'http://localhost:8080/superx/servlet/SuperXmlTabelle?tid=19220&kennung=superx&passwort=anfang12' pruefprotokoll.xls
```

Die Datei wird im gleichen Verzeichnis gespeichert.

> Sicherheitshinweis: wenn Sie Passworte im Klartext in Browser-Adressleisten oder in Login-Shells eintippen, werden diese an verschiedenen Stellen gespeichert. Im Browser ist es die Chronik bzw. der Cache, in der Shell ist es die Eingabehistorie. Dies macht es anderen Anwendern leicht, die Passworte auszuspähen. Sie sollten daher die Passworte in Shellscripte verlagern, die ohne Login-Shell ablaufen. Diese Shellscripte wiederum dürfen nicht von unbefugten Personen eingesehen werden, stellen Sie die Leserechte im Dateisystem entsprechend her..

### 2.4.4Mailversand von Berichtsausgaben

Im Tandem mit dem obigen Berichtsdownload können Sie auch Berichtsausgaben per Mail versenden. Dafür sind im Pakt Scripte zum Verschicken von Dateien per Email

```
sendmail.bat  (für Windows)
sendmail.x    (für Linux)
```

Es sind die folgenden Parameter vorgesehen:

```
sendmail.bat --to test@test.de --from system@super-ics.de --host smtp.strato.de --ssl (optional wenn SSL verwendet werden soll) --username system --password geheim --subject "COB Prüfprotokoll"  --msg "Hier erhalten Sie Ihre Protokolle" (optional) --msgfile c:\nachricht.txt (optional) --attach c:\protokoll.xls (optional)
```

Die Parameter sind selbsterklärend. Der Parameter "--subject" kennzeichnet die Betreffzeile, und in den Mailtext selbst kann man "--msg Nachricht" oder mit "--msgfile Dateiname" auch den Inhalt von Textdateien kopieren. Außerdem wird mit dem Parameter "--attach Dateiname" eine Datei angehängt.

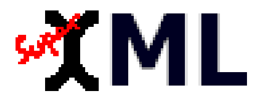
# 3Administration des Kernmoduls: HowTo

Im folgenden werden zentrale Arbeitsschritte beim Betrieb von SuperX beschrieben. Für einen Blick auf den Hintergrund sollten Sie sich ggf. die Bestandteile anschauen.

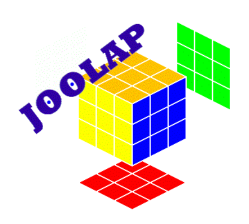
Zuächst zeigen wir, wie die Frontends funktionieren, und dann beschreiben wir die Werkzeuge für die Administration von SuperX.

SuperX verfügt über unterschiedliche Benutzeroberflächen, hier "Frontends" genannt. Das **SuperX-Applet** dient dem allgemeinen Berichtswesen und liefert vordefinierte Ergebnistabellen. Die Installati-

on des Applets auf den Clients ist in der Installationsanleitung beschrieben. Die Funktionsweise des Applets ist ausführlich in dem Benutzerhandbuch dokumentiert.




Das XML-Frontend liefert komplexe Berichte, die aus mehreren Ergebnistabellen zusammengestellt werden, und die flexibel für verschiedene Ausgabegeräte und –formate aufbereitet werden können. Im Gegensatz zum Applet sind keinerlei Installationsschritte notwendig, es genügt ein html4-fähiger Browser. Derzeit ist das XML-Frontend noch im Betastadium.




**Joolap** bietet die Möglichkeit, multidimensionale Auswertungen zu machen und Statistiken flexibel den eigenen Bedürfnissen anzupassen. Joolap wird mit einer eigenen Dokumentation ausgeliefert.

## 3.1Die SuperX-Administrationswerkzeuge

Die Verwaltung des Organigramms und des Themenbaums sowie grundlegende User- und Gruppenverwaltung lässt sich mit Hilfe eines graphischen Administrationswerkzeugs SuperXAdmin (Betaversion 1.0) sowie über ein Access-Frontend erledigen. Es gibt neben der Shell-Zugang über UNIX zwei Administrationswerkzeuge für das Kernmodul: Browser-basierte Formulare im XML-Frontend, die auf die DB-FORMS-Technologie zurückgreifen. Außerdem wurde ein Access-Frontend entwickelt, dass über ODBC-Verknüpfung einen direkten Zugriff auf die SuperX-Tabellen liefert. Das Browser-basierte Frontend hat den Vorteil, dass es auch über eine http-Verbindung arbeitet und somit höhere Sicherheitsstandards erfüllt. Das Access-Frontend eignet sich besser für die direkte Bearbeitung einzelner Tabellen und für die Entwicklung von Abfragen. Die Funktionalität ist ansonsten identisch, deshalb wird im folgenden nur die Oberfläche des Browser-Frontends beschrieben. Lediglich die Abfragenbearbeitung mit dem Access-Frontend wird gesondert dargestellt.

### 3.1.1Übersicht über Scripte unter UNIX

#### 3.1.1.1Allgemeine Prozessverwaltung

Mit folgenden UNIX-Kommandos können Sie die Auslastung des Servers feststellen:

| free -m | Zeigt den genutzten Arbeitsspeicher an |
|---|---|
| top | Zeigt die Prozesse und deren Prozessor/RAM-Auslastung an |
| ps fax \| grep superx | Zeigt an welche Prozesse überhaupt laufen (unter AIX:<br>ps -ef \| grep superx<br>) |
| kill -9 <<ProzeßID>> | Beendet einen Prozeß |

### 3.1.1.2SuperX-spezifische Scripte: Übersicht

Für die Administration des DataWarehouse sind Shellscripte vorbereitet, die flexible Werkzeuge zur Datenbankadministration bereitstellen. Die Shellscripte werden in den Update-Scripten aufgerufen, können aber auch zur manuellen Administration benutzt werden. Die wichtigsten Bereiche sind die Masken-Verwaltung und die Ladescripte im Umgang mit Tabellen sowie allgemeine Scripte.

Alle Scripte befinden sich unter `$SUPERX_DIR/db/bin`, deshalb muss dieser Pfad inder Umgebungsvariable PATH enthalten sein. Die Scripte wurden unter UNIX entwickelt (ohne Endung oder Endung `.x`), einige davon sind auch nach DOS portiert worden (erkennbar an der Endung `.bat`).

Einige Scripte lauten "sx_auto_"..., dies bedeutet, dass die Scripte ohne Sicherheitsabfrage ausgeführt werden.

Voraussetzung für den Ablauf der Scripte ist die Eintragung der korrekten Umgebungsvariablen in `$SUPERX_DIR/db/bin/SQL_ENV` bzw. `$SUPERX_DIR\db\bin\sql_env.bat`. Wenn der Client `jdbc` verwendet wird, muss ausserdem die korrekte DB_PROPERTIES gesetzt sein.

### 3.1.1.3Die Umgebungssteuerung: SQL_ENV

Das Script `$SUPERX_DIR/db/bin/SQL_ENV` steuert die Umgebung und ist für den Betrieb der Scripte unverzichtbar. Einige Variablen sind vorbelegt, Beispiele sind auf Kommentar gesetzt. Da die Umgebung von dem System abhängt, muss jeder Anwender die Werte manuell pflegen. Bei einem Update des SuperX-Kernmoduls wird diese Datei nicht überschrieben, lediglich sein SQL_ENV.sam im gleichen Verzeichnis. Von dort müssen relevante Änderungen dann in die "richtige" SQL_ENV manuell übernommen werden. Informix- und Postgres-spezifische Variablen sind in dem Kapitel zur Installation und Konfiguration der Datenbankserver beschrieben.

Folgende Variablen sind auf jeden Fall zubelegen:

| | |
|---|---|
| **SUPERX_DIR** | Der Installationspfad von SuperX |
| **DATABASE** | Das Datenbanksystem ("POSTGRES" / "INFORMIX") |
| **DBNAME** | Der Name der Datenbank (standardmäßig "superx"). |
| **SX_CLIENT** | Die Clientanwendung (bei Postgres "psql", bei Informix "dbaccess"). Ein client namens "jdbc" ist generisch und dient dem Zugriff auf beliebige DB-Systeme, für die jdbc-Treiber existieren.<br>Der jdbc-Client wurde bisher mit Informix, Postgres und hsqldb getestet – die jdbc-Treiber für Informix und Postgres werden mitgeliefert und dürfen auf keinen Fall durch andere ersetzt werden. |
| **MAILPROG** | Das Mailprogramm unter UNIX, z.B. `mutt` oder `mail`; dies muss sich im PATH des users superx befinden. |
| **LOGMAIL/ERRORMAIL** | Die superx-weite Mailadresse, an die Logdateien von ETL-Scripten geschickt werden. |
| **JAVA_HOME** | Installationspfad der Java-Runtime. Das Unterverzeichnis bin muss in den PATH aufgenommen werden. |
| **JAVA_OPTS** | Java-Runtime-Optionen, z.B. RAM<br>Bei Einsatz unter Cygwin muss ggf. folgende Einstellung gemacht werden:<br>`` XALAN_PATH=`cygpath --path --windows ``<br>`` "$JAVA_HOME"/jre/lib/endorsed/xalan2-6-0.jar` ``<br>`JAVA_OPTS="-Xmx200M -Djava.awt.headless=true`<br>`-Xbootclasspath/p:"$XALAN_PATH` |
| **DB_PROPERTIES** | Der Pfad zur DB_PROPERTIES, standardmäßig<br>`$SUPERX_DIR/webserver/`<br>`tomcat/webapps/superx/WEB-INF/db.properties` |
| **MANDANTID** | Mandantennummer (Hochschulnummer) bei mandantefähigen Installationen |

Die folgenden Umgebungsvariablen sind nur für den JDBC-Client sowie für Postgres relevant:

| | |
|---|---|
| **LOGGING_PROPERTIES** | Logging-Parameter für den jdbc-Client. Voreingestellt ist "WARNING", mehr Ausgaben erhält man mit "FINE" |
| **PG_HOST** | Name des Postgres-Servers (für Postgres unter Windows) |

Die folgenden Umgebungsvariablen werden wahrscheinlich nicht geändert (sollten sie auch nicht):

| | |
|---|---|
| **DBDELIMITER** | Standardmäßig "^" |
| **PATH** | Der PATH wird erweitert um das Verzeichnis<br>`./:$SUPERX_DIR/db/bin` |
| **JDBC_CLASSPATH** | Der Pfad zu den relevanten jdbc-Treibern und Hilfsprogrammen. |
| **XML_CLASSPATH** | Der Pfad zu den XML-Tools (Xalan, Xerces & co). |

Die Datei sollte unter UNIX in jedem Aufruf der shell "gesourced" werden, z.B. durch den Befehl:

```
. ~/db/bin/SQL_ENV
```

(Leerzeichen zwischen Punkt und Tilde!) in der Datei `~/.bashrc`.

Wenn Sie unter Windows den jdbc-Client nutzen, dann müssen Sie die Datei als erstes in der DOS-Shell aufrufen, bzw. in definierten Tasks am Anfang aufrufen.

### 3.1.1.4Allgemeine Scripte

DOSQL

DOQUERY

sx_transform

Propadmin

Zum Absetzen beliebiger SQL-Kommandos werden die Befehle DOSQL und DOQUERY genutzt.

#### DOSQL

Shellvariablen setzen und SQL-Anweisung(en) in der der Datei (als Parameter) in der SuperX-Datenbank ausfuehren.

**Syntax**

```
DOSQL "Dateiname mit sql-Anweisung(en)" header (true,false)
(optional) Ausgabedatei(optional)
```

**Beispiel**

```
DOSQL "/home/superx/db/isql/test.sql" false "^" output.txt
```

Das Ergebnis kann mit Feldüberschriften (header=true) in eine Datei Ausgabedatei ausgegeben werden

#### DOQUERY

Shellvariablen setzen und eingegebene SQL-Anweisung (als Parameter) in der SuperX-Datenbank ausfuehren.

**Syntax**

```
DOQUERY "sql-Anweisung" header (true,false)(optional) Delimi-
ter(optional) Ausgabedatei(optional)
```

**Beispiel**

```
DOQUERY "select name from userinfo" false "^" output.txt
```

Das Ergebnis kann mit Feldüberschriften (header=true) in eine Datei Ausgabedatei ausgegeben werden.

#### sx_transform

Transformiert eine xml-Datei mit einer übergebenen XSL-Datei und gibt das Ergebnis in einen Ausgabekanal aus (stdout oder Datei). Dabei wird der in SuperX integrierte XML-Parser Xerces und der XML-Prozessor Xalan benutzt.

**Syntax**

```
sx_transform.x IN:<xml-Datei> -XSL:<xsl-Datei> -OUT:<Ausgabe-
datei> -method:<Ausgabeformat (text, xml,html,rtf,pdf)>(optio-
nal) <Parameter>(optional)
```

**Beispiel**

```
sx_transform.x -IN:myxml.xml -XSL:myxsl.xsl -OUT:output.htm
-method:html
```

Als Parameter kann ein spezielles Ausgabeformat gewählt werden, z.B. TEXT (siehe Xalan-Doku). Bei rtf wird der RTF-Construktor Jfor auferufen, bei pdf wird FOP aufgerufen. Die *.fo-Datei wird nach tmp.-fo geschrieben und dann nach pdf transformiert. Wir gehen also nich davon aus, dass .fo-Dateien die Eingabequellse darstellen.

#### Propadmin

Der PropAdmin ist ein kleines Werkzeug, um den Zugriff auf jdbc-Datenbanken zu testen und die Verbindungsparameter in einer übergebenen properties-Datei zu sichern. Wenn keine graphische Umgebung eingerichtet ist, müssen Sie die alle Verbindungsparameter manuell in die db.properties eintragen. Nur das Passwort kann mit dem propadmin bearbeitet werden.

(Musterdateien für Postgres und Inofrmix liegen vor in `$SUPERX_DIR/webserver/tomcat/webapps/superx/WEB-INF/db-postgres.properties` bzw. `db-informix.properties`). Wenn als weiterer Parameter kein Dateiname übergeben wird, dann wird die Umgebungsvariable DB_PROPERTIES ausgewertet.

| | |
|---|---|
| **Syntax UNIX** | `propadmin.x -nogui(optional) <Dateipfad zu db.properties>(optional)` |
| **Syntax DOS** | `propadmin.bat <Dateipfad zu db.properties>(optional)` |

Wenn die Default-Dateiencodierung der aktiven Locale für die Passwort-Verschlüsselung nicht ausreicht, wird eine Fehlermeldung ausgegeben. Unter Windows / DOS ist die Vorbelegung Cp1252 bei deutscher Codepage ausreichend, unter Unix wird die deutsche Locale benötigt.

### 3.1.1.5 Codierung in ISO und UTF-8

sx_show_encoding.x
sx_recode_iso2utf.x
sx_recode_utf2iso.x
sx_list_isofiles.x
sx_recode_isofiles.x
sx_list_utf8files.x
sx_recode_utf8files.x

Ältere Systeme arbeiten in der Regel mit der Zeichencodierung `ISO-8859-1` bis `ISO-8859-9`. Dieser Zeichensatz wird auch LATIN-1 genannt. Die UNIX-Locale `de_DE@euro` entspricht z.B. `ISO-8859-9` und enthält das EUR-Symbol.

Mit dem Wechsel von ISO-Codierung zu UTF8 bleibt oft der Bedarf bestehen, Dateien hin- und herzucodieren. Seit es, weil beim Entladen aus einer entfernten Datenbank noch das ISO-System genutzt wird, oder bei der Migration eines Systems. Nach unserer Erfahrung sollten Umlaute in Dateinamen unbedingt vermieden werden.

SuperX bietet unter UNIX Shellscripte zur Erfassung und Änderung der Zeichencodierung (Verzeichnis $SUPERX_DIR/db/bin). Im Wesentlichen werden dabei die Unix-Kommandos `file` und `recode` genutzt, die Shellscripte machen den Umgang mit umfangreichen Dateilisten komfortabler. Bei der Verarbeitung von Dateilisten sollte man die Scripte sehr vorsichtig einsetzen, es finden keine Sicherheitsüberprüfungen statt.

| | |
|---|---|
| **Achtung: nur unter Linux getestet** | Die Scripte wurden bisher nur unter Linux getestet, andere UNIXe wie Solaris und AIX11 verhalten sich ggf. anders als erwartet. Daher bitte mit Vorsicht benutzen. |

**sx_show_encoding.x**

Das Script zeigt die Encodierung einer Datei an.

| | |
|---|---|
| **Syntax** | `sx_show_encoding.x <Datei>` |
| **Beispiel** | `sx_show_encoding.x $SUPERX_DIR/webserver/tomcat/webapps/superx/WEB-INF/web.xml` |
| **Ausgabe** | `/hopme/superx/webserver/tomcat/webapps/superx/WEB-INF/web.xml:`<br>`XML`<br>`ISO` |

Das Script nutzt verschiedene UNIX-Tools, je nach System kann die Ausgabe variieren. Bei XML-Dateien wird auch der Dateiinhalt (XML-Header) ausgewertet.

### sx_recode_iso2utf.x

Das Script ändert die Encodierung einer Datei von ISO nach UTF-8:

| | |
|---|---|
| **Syntax** | `sx_recode_iso2utf.x <Datei>` |
| **Beispiel** | `sx_recode_iso2utf.x $SUPERX_DIR/webserver/tomcat/webapps/superx/WEB-INF/web.xml` |
| **Ausgabe** | `--keine--` |

Das Script nutzt das UNIX-Kommando `recode`. Darüberhinaus werden bei XML-Dateien auch die XML-Header "encoding=..." geändert, so wird z.B. aus

```
<?xml version="1.0" encoding="ISO-8859-1"?>
```

der Header

```
<?xml version="1.0" encoding="UTF-8"?>
```

Andere Inhalte der Datei unterhalb der ersten Zeile werden keinesfalls geändert.

### sx_recode_utf2iso.x

Das Script ändert die Encodierung einer Datei von ISO nach UTF-8:

| | |
|---|---|
| **Syntax** | `sx_recode_utf2iso.x <Datei>` |
| **Beispiel** | `sx_recode_utf2iso.x $SUPERX_DIR/webserver/tomcat/webapps/superx/WEB-INF/web.xml` |
| **Ausgabe** | `--keine--` |

Das Script nutzt das UNIX-Kommando `recode`. Darüberhinaus werden bei XML-Dateien auch die XML-Header "encoding=..." geändert, so wird z.B. aus

```
<?xml version="1.0" encoding="UTF-8"?>
```

der Header

```
<?xml version="1.0" encoding="ISO-8859-1"?>
```

Andere Inhalte der Datei unterhalb der ersten Zeile werden keinesfalls geändert.

### sx_list_isofiles.x

Das Script listet alle ISO-Dateien im übergebenen Verzeichnis auf (inkl. Unterverzeichnisse).

**Syntax** | sx_list_isofiles.x <Pfad>
**Beispiel** | sx_list_isofiles.x webserver/tomcat/webapps/superx/WEB-INF
**Ausgabe** | webserver/tomcat/webapps/superx/WEB-INF/lib/LocalStrings_de.properties
webserver/tomcat/webapps/superx/WEB-INF/lib/hierhin_den_informix_treiber_kopieren.txt
[...]
webserver/tomcat/webapps/superx/WEB-INF/db.properties

Die Ausgabe kann in eine Datei umgeleitet werden, welche wiederum für das Script sx_recode_isofiles.x als Eingabedatei genutzt werden.

```
sx_list_isofiles.x webserver/tomcat/webapps/superx/WEB-INF >iso.txt
```

**sx_recode_isofiles.x**

**Das Script konvertiert alle Dateien in der übergebenen Dateiliste von ISO nach UTF-8:**

**Syntax** | sx_recode_isofiles.x <Datei>
**Beispiel** | sx_list_isofiles.x iso.txt
**Ausgabe** | --Keine--

Die Eingabedatei ist in der Regel die Ausgabe des Scriptes sx_list_isofiles.x.

**sx_list_utf8files.x**

Das Script listet alle UTF-8-Dateien im übergebenen Verzeichnis auf (inkl. Unterverzeichnisse).

**Syntax** | sx_list_utf8files.x <Pfad>
**Beispiel** | sx_list_utf8files.x webserver/tomcat/webapps/superx/WEB-INF
**Ausgabe** | webserver/tomcat/webapps/superx/WEB-INF/lib/LocalStrings_de.properties
webserver/tomcat/webapps/superx/WEB-INF/lib/hierhin_den_informix_treiber_kopieren.txt
[...]
webserver/tomcat/webapps/superx/WEB-INF/db.properties

Die Ausgabe kann in eine Datei umgeleitet werden, welche wiederum für das Script sx_recode_utf8files.x als Eingabedatei genutzt werden.

```
sx_list_isofiles.x webserver/tomcat/webapps/superx/WEB-INF >utf.txt
```

**sx_recode_utf8files.x**

Das Script konvertiert alle UTF-8 Dateien in der übergebenen Dateiliste von UTF-8 nach ISO:

**Syntax** | sx_recode_utf8files.x <Datei>
**Beispiel** | sx_recode_utf8files.x utf.txt
**Ausgabe** | --Keine--

Die Eingabedatei ist in der Regel die Ausgabe des Scriptes sx_list_utf8files.x.

### 3.1.1.6Umgang mit Tabellen

sx_unload_table
sx_upload_table
sx_upload_records
sx_schema

In SuperX werden ständig Tabellen erstellt / geladen / entladen. Zu diesem Zweck wurden Shellscripte entwickelt.

#### sx_unload_table

Entlädt die Inhalte der Tabelle nach <<Dateiname>>(optional) oder <<name>>.unl

| | |
|---|---|
| **Syntax** | sx_unload_table.x <<name>> <<Dateiname>>(optional) |
| **Beispiel** | sx_unload_table.x userinfo |

#### sx_upload_table

Löscht die Inhalte der Tabelle <<name>>, und lädt die Inhalte einer Datei in die Tabelle mit sx_upload_records. Wenn kein Dateiname übergeben wurde, wird als Name <<name>>.unl angenommen.

| | |
|---|---|
| **Syntax** | sx_upload_table.x <<name>> <<Dateiname>>(optional) |
| **Beispiel** | sx_upload_table.x userinfo |

#### sx_upload_records

Lädt die Inhalte einer Datei in die Tabelle, ohne vorherige Inhalte zu löschen. Wenn kein Dateiname übergeben wurde, wird als Name <<name>>.unl angenommen.

| | |
|---|---|
| **Syntax** | sx_upload_records.x <<name>> <<Dateiname>>(optional) |
| **Beispiel** | sx_upload_records.x userinfo |

Bei Postgres als DB-System wird eine Java-Klasse (de.superx.bin.UnlFileConverter) aufgerufen, die die Unload-Datei entsprechend einer Spezifikation aufbereitet (siehe $SUPERX_DIR/db/conf/unldescr*).

Wenn der jdbc-Client benutzt wird, können umfangreiche Parameter übergeben werden (Import mit Spaltenüberschriften, Ausgabe von Fehlerprotokollen). Vergleichen Sie die Kommentare im Script.

#### sx_schema

Entlädt das Schema einer Tabelle in einem vorgegebenen Format.

| | |
|---|---|
| **Syntax** | `sx_schema.x sx_schema <Tabelle> <format (pg|ids|ansi|xml|HIS))>(optional) <Ausgabedatei> (optional)` |
| **Beispiel** | `sx_schema.x userinfo ids myschema.sql` |
| **Die Formate** | Die Formate sind entweder sql-Scripte für die jeweiligen Datenbanktypen (Postgres, Informix, ANSI), die aus der Umgebungsvariable DATABASE ausgelesen werden, oder xml bzw. ein xml-Format in Anlehnung an die Datenbank-DTD der HIS GmbH. |

### 3.1.1.7Modulverwaltung

Bisherige SuperX-Implementationen sind an den Hochschulen entstanden und haben dementsprechend eine große Vielfalt von Update-Scripten, die jeweils die Vorlieben und Bedingungen der jewieligen Hochschule wiederspiegeln. Daraus ergibt sich für "Neulinge" ein sehr verwirrendes Bild. Außerdem gestaltet sich der Entwurf eines Moduls recht aufwändig, weil die ETL-Funktionen (Extraction -> Transformation -> Loading) manuell programmiert werden müssen. Eine weitere Anforderung ist, daß SuperX auf zwei Datenbankplattformen lauffähig sein muss, Informix und Postgres.

Das Ergebnis ist: SuperX ist (auf Datenbankseite) sehr fehleranfällig, schwer wartbar und praktisch nicht updatebar.

Mit SuperX Version 2.1 wurde die Verwaltung der Module (Installieren / Aktualisieren / sichern und die zugehörogen Logdateien) in zentrale Shellscripte verlagert, die sich ebenfalls in `$SUPERX_DIR/db/bin` befinden. Die Shellscripte sind dabei nur die operativen "Hüllen" um die eigentlichen SQL-Scripte. Diese wiederum werden zum Teil "von Hand" erzeugt (um z.B. hochschulspezifische Erweiterungen oder Anpassungen vorzunehmen), und zum Teil automatisch aus einer zentralen Steuerdatei (`$SUPERX_DIR/db/module/<<Modulname>>/conf/<<Modulname>>.xml`) jeweils für Postgres und Informix erzeugt.

#### 3.1.1.7.1module_scripts_create.x

Das Script erzeugt die Installationsdateien für ein Modul, jeweils für Postgres und Informix, nach dem Schema

`<<Modulname>>_<<Scriptaktion>>_<<Kürzel für Datenbanksystem>>.sql`

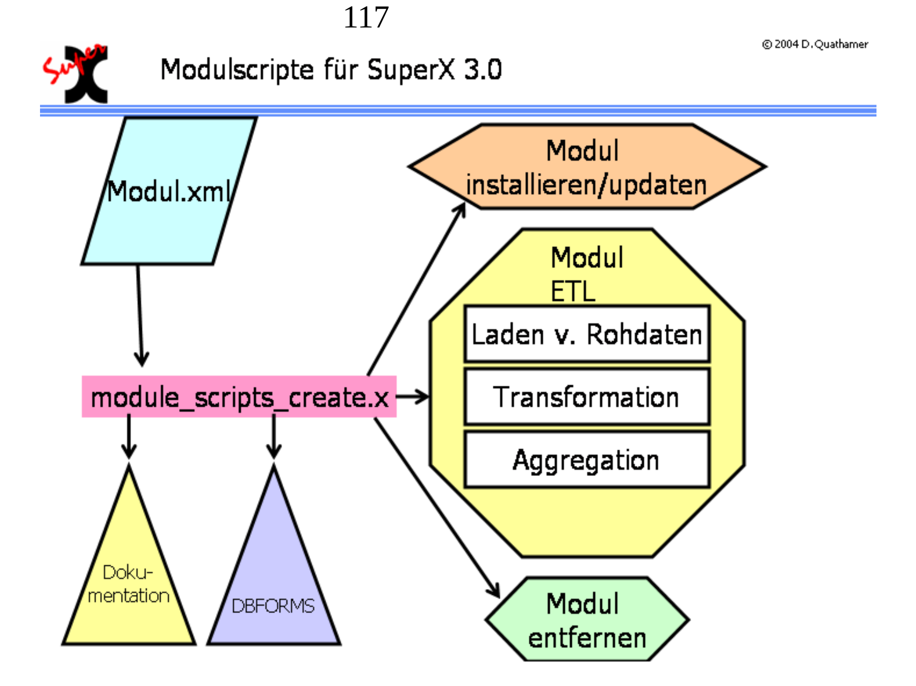
Z.B. wird für das BAU-Modul aus der Datei `$BAU_PFAD/conf/bau.xml` das Script `bau_load_pg.sql` erzeugt, das die Rohdaten unter Postgres lädt, oder die Datei `bau_trans_ids.sql` für das Script, das die Bau-Tabellen unter Informix transformiert

| | |
|---|---|
| **Syntax** | `module_scripts_create.x <<Modulename>> <<Modulpfad>> <<Datenbanksystem(optional)>> <<Versionsnr.(optional)>>` |
| **Beispiel** | `module_scripts_create.x BAU $BAU_PFAD INFORMIX 1.0` |

Im Grunde handelt es sich um XML-Transformationen. Die Stylesheets für dieses Script befinden sich im Verzeichnis `$SUPERX_DIR/db/conf`, und die XML-Datei für das Module in `$SUPERX_DIR/db/module/<<Modulname>>/conf`. Wenn die Datei nicht gefunden wird, bricht das Script ab.

Die folgende Abbildung zeigt die Arbeitsweise:

Das Script module_scripts_create.x erzeugt eine Reihe von Scripten, die das Modul installieren / aktualisieren / deinstallieren. Außerdem werden html- bzw. rtf-Dokumentationen erzeugt sowie Administrationsformulare für **dbforms**.




### 3.1.1.7.2 module_install.x

Installiert ein Modul <<Modulname>> in der Datenbank, wobei die Installationsdateien sich im <<Modulpfad>> befinden.

| | |
|---|---|
| **Syntax** | module_install.x <<Modulename>> <<Modulpfad>> |
| **Beispiel** | module_install.x BAU $BAU_PFAD |

### 3.1.1.7.3 module_drop.x

Löscht die Komponenten eines Moduls <<Modulname>> in der Datenbank, wobei die Installationsdateien sich im <<Modulpfad>> befinden.

| | |
|---|---|
| **Syntax** | module_drop.x <<Modulename>> <<Modulpfad>> |
| **Beispiel** | module_drop.x BAU $BAU_PFAD |

### 3.1.1.7.4 Entladen

Das Entladescript lautet $SUPERX_DIR/db/module/<<Komponentenname>>/rohdaten/<<Komponentenname>>_unload.x. Die Entladeparameter werden in folgender Datei festgelegt:
$SUPERX_DIR/db/module/<<Komponentenname>>/rohdaten/<<Komponentenname>>_ENV

Entladeroutine bei mandantenfähigen Installationen:
$SUPERX_DIR/db/module/<<Komponentenname>>/rohdaten<<Mandantid>>/<<Komponentenname>>_ENV

Dokumentation zu den jew. Parametern finden Sie in den jeweiligen Administrationshandbüchern der Module. Meist kann man Start-Semester oder -Jahre für das Entladen festlegen. Immer muß man auch das Datenbank-Vorsystem festlegen (Hostname, Kennung etc) sowie, bei HIS-Systemen, die Versionsnummer.

### 3.1.1.7.5module_update.x

Installiert eine neue Version eines Moduls <<Modulname>> in der Datenbank, wobei die Installationsdateien sich im <<Modulpfad>> befinden.

| | |
|---|---|
| **Syntax** | module_update.x <<Modulename>> <<Modulpfad>> |
| **Beispiel** | module_update.x BAU $BAU_PFAD |

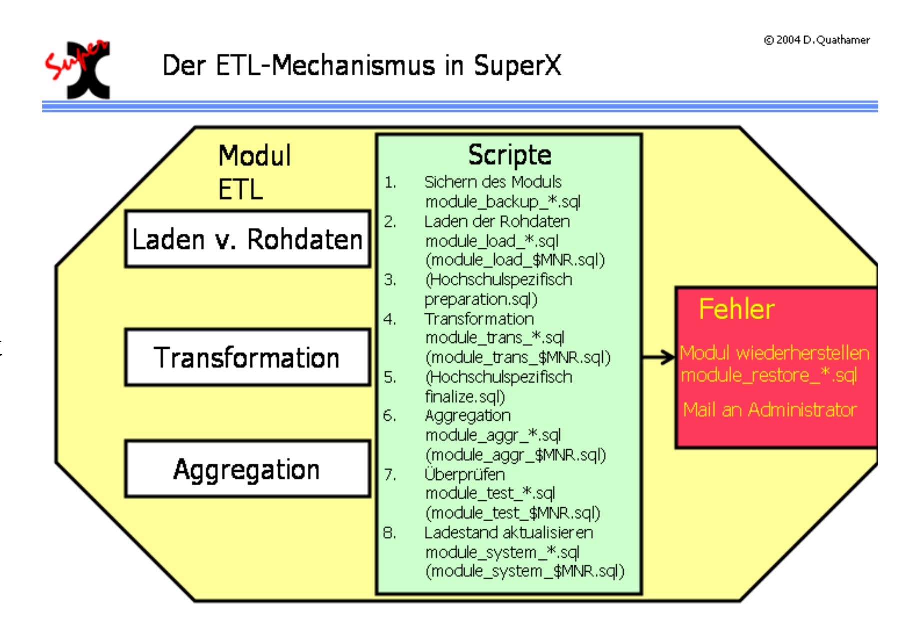
### 3.1.1.7.6module_etl.x

Aktualisiert ein Modul <<Modulname>> in der Datenbank, wobei die Installationsdateien sich im <<Modulpfad>> befinden.

| | |
|---|---|
| **Syntax** | module_etl.x <<Modulename>> <<Modulpfad>> |
| **Beispiel** | module_etl.x bau $BAU_PFAD |

Die folgende Abbildung zeigt, wie die Komponenten zusammenhängen (klicken Sie auf die Grafik, um sie zu vergrößern):

Das Modul wird zunächst nach $MODULPFAD/tmp gesichert, danach werden die Rohdaten geladen, die Daten vorbereitet, transformiert, und nachbereitet. Danach werden die Hilfstabellen erzeugt und, bei erfolgreichem durchlaufen, das Standdatum aktualisiert.




Bei Fehlern im ETL-Prozeß wird die Sicherung wiederhergestellt, und eine Mail an den Administrator verschickt. Außerdem werden die übernommenen Daten überprüft; wenn z.B. Schlüssel fehlen oder Rohdaten falsch zu sein scheinen, wird dies als Attachment an die Log- oder Fehlermail angehängt.

In der Praxis wird dieses Script nicht direkt von Cronjobs ausgeführt, sondern von einem Shellscript, das vorher die Umgebung einrichtet. Das folgende Beispiel zeigt das Update-Script für Bau unter Informix:

**bau_update.x**

```
#!/bin/sh
. /home/superx/db/bin/SQL_ENV
DBMONEY=,
export DBMONEY
ERRORMAIL="bau-admin@hochschule.de"
export ERRORMAIL
LOGMAIL=$ERRORMAIL
export LOGMAIL
module_etl.x bau $BAU_PFAD >$BAU_ERRORDAT 2>&1
```

Weil Buisy mit "," als Dezimaltrenner arbeitet, wird ausnahmsweise DBMONEY auf "," gesetzt. Außerdem ist es möglich, für jedes Modul unterschiedliche Mailadressen zuzuweisen. Die Mailadressen werden in der SQL_ENV eingetragen.

Im allgemeinen ETL-Prozeß wird standardmäßig auch die Tabelle `protokoll` in einem festzulegendem Rhythmus (Konstante `Löschung Protokoll (Tage)`) gelöscht. Beim Vorgabewert 90 werden bei jeder ETL-Routine Einträge, die älter als 90 Tage sind, gelöscht.

###### *3.1.1.7.6.1Hochschulspezifische Transformationen im ETL-Prozeß*

Zusätzlich lassen sich im ETL-Prozeß hochschulspezifische Scripte ausführen (und überwachen). Dazu müssen Dateien mit einem gewissen Namensschema im Stammverzeichnis des Moduls vorhanden sein. Es gibt einen vereinfachten und einen erweiterten Modus für hochschulspezifische Transformationen.

**Einfacher Modus**

Wenn im Modulpfad die Datei "preparation.sql" existiert, wird sie nach dem LOAD-Schritt ausgeführt.
Wenn im Modulpfad die Datei "finalize.sql" existiert, wird sie nach dem TRANS-Schritt ausgeführt.

**Erweiterter Modus: Mandantenspezifische Scripte**

Wenn im Modulpfad Dateien nach dem Schema
<<Modulname>>_<<ETL-Schritt>>_<<Mandatennr.>>.sql
exisitieren,w erden diese jeweils nach dem "normalen" ETL-Schritt ausgeführt.
Wenn also z.B. die Datei
`cob_trans_70.sql`
existiert und in der `SQL_ENV` die Umgebungsvariable `$MANDANTID` auf "70" steht, dann wird das Script nach dem normalen Trans-Schritt ausgeführt und nach `L_cob_trans_mandant_70.log` geloggt.

Der erweiterte Modus erlaubt die beliebige Anpassung eines Modus an eigene Bedürfnisse, z.B. Schlüsselumsetzung o.ä. Gleichzeitig erlaubt er einen echten mandentenfähigen Betrieb der ETL-Scripte.

##### 3.1.1.7.7Logging der Shellscripte

Hinweis: bei mandantenfähigen Installationen steht vor der Endung `.log` immer die MandantID.

##### *3.1.1.7.7.1Installation / Upgrade*

Fürs Kernmodul lauten die Dateien bei der Installation:
$SUPERX_DIR/db/install/L_*.log
Beim Upgrade:
$SUPERX_DIR/db/install/upgrade.log
Für alle anderen Komponenten:
$SUPERX_DIR/db/module/<<Komponentenname>>/L_<<Komponentenname>>_<<Installationsschritt>>.log

##### *3.1.1.7.7.2Laderoutinen*

Für alle Module sind die Dateien wie folgt benannt:
Entladeroutine:
$SUPERX_DIR/db/module/<<Komponentenname>>/rohdaten/<<Komponentenname>>_unload.err
Entladeroutine bei mandantenfähigen Installationen:
$SUPERX_DIR/db/module/<<Komponentenname>>/rohdaten<<Mandantid>>/<<Komponentenname>>_unload.err

Laderoutine:
$SUPERX_DIR/db/module/<<Komponentenname>>/L_<<Komponentenname>>_<<Ladeschritt>>.log

- wobei <<Ladeschritt>> wie folgt aufgebaut ist:
- 1. Unload (Entladen aus Vorsystem)
- 2. Load (CSV-Upload)
- 3. Transformation (Schlüsselharmonisierung, Prüfroutinen)
- 4. Aggregation (Aufbau der Hilfstabellen)
- 5. System (Ladedatum aktualisieren, Datenbank-Wartung)

Wenn die Laderoutine erfolgreich ist, werden alle Schritte hintereinander ausgeführt und geloggt. Wenn nicht, dann wird der jew. Schritt zuende geführt, und  dann die Laderoutine gestoppt. Wenn also z.B. beim LOAD ein Fehler auftritt, dann wird der Schritt "Transformation" gar nicht erst begonnen. So ist sichergestellt daß die Auswertungen trotz Fehler laufen.

#### 3.1.1.8Masken-Verwaltung

Die Masken-Verwaltung ist detailliert im **Entwicklerhandbuch SuperX** beschrieben. Hie rnur ein paar Hinweise zur Verwaltung der Masken. Zum Erzeugen und Verändern von Masken gibt es unter UNIX eine Kommandoschnittstelle, die auf dem Gebrauch folgender Skripte beruht. Die Skripte stehen unter dem Verzeichnis
$SUPERX_DIR/db/masken
und erzeugen oder verwenden Dateien in dem gegenwärtigen Arbeitsverzeichnis. Nach dem Einspielen der Datenbank sollten Sie darauf achten, den Dateien Ausführungsberechtigung (chmod 750 sx_*) zu geben.

#### 3.1.1.8.1Eine Maske suchen

Wenn Sie eine Maske suchen, sollten die die Felder tid oder name in der Tabelle maskeninfo durchsuchen. Das folgende Script macht dies automatisch:

**sx_search_mask**

| Aufruf: | sx_search_mask <String> |
|---|---|
| Aktion: | sx_search_mask sucht die Masken, deren Name <String> enthält |
| Ausgabe: . | tid, name der gefundenen Masken |

#### 3.1.1.8.2Eine Maske sichern und entladen

Um eine Maske zu sichern, müssen Sie die entsprechenden Einträge in den Tabellen

1. `felderinfo`,
2. `masken_felder_bez`,
3. `maskeninfo`,
4. `sachgeb_maske_bez`,
5. `maske_system_bez`

selektieren und sichern. Für dies gibt es das Script `sx_select_mask`.

**sx_select_mask**

| Aufruf: | `sx_select_mask <TID>` |
|---|---|
| Aktion: | `sx_select_mask` entlädt alle Metadaten aus den Tabellen maskeninfo, felderinfo, masken_felder_bez, sachgeb_maske_bez, maske_system_bez zur Maske mit tid = <TID>. |
| Ausgabe: | Fünf Dateien:<br>1.<TID>_felderinfo.unl,<br>2.<TID>_masken_felder_bez.unl,<br>3.<TID>_maskeninfo.unl,<br>4.<TID>_sachgeb_maske_bez.unl,<br>5.<TID>_maske_system_bez.unl |

#### 3.1.1.8.3Eine Maske neu einfügen

Um eine Maske neu einzufügen, müssen Sie die entsprechenden Einträge in den Tabellen

1. `felderinfo`,
2. `masken_felder_bez`,
3. `maskeninfo`,
4. `sachgeb_maske_bez`,
5. `maske_system_bez`

einfügen. Dafür gibt es das Script `sx_insert_mask`.

**sx_insert_mask**

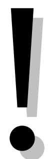
| Aufruf: | `sx_insert_mask <TID> [<neue TID>] [j]` |
|---|---|
| Aktion: | sx_insert_mask lädt den Inhalt der fünf Dateien<br>1.<TID>_felderinfo.unl, |

| | |
|---|---|
| | 2.<TID>_masken_felder_bez.unl,<br>3.<TID>_maskeninfo.unl,<br>4.<TID>_sachgeb_maske_bez.unl,<br>5.<TID>_maske_system_bez.unl<br>in die jeweiligen Tabellen der SuperX-Datenbank. Mit "j" wird die Sicherheitsabfrage umgangen. |

Falls <neue TID> nicht angegeben wird, werden die Metadaten wieder mit der alten TID in die Datenbank eingespielt (=Update).

Falls <neue TID> angegeben wird, werden die Metadaten mit der neuen TID in die Datenbank eingespielt (=Insert). Dabei werden alle TIDs in den abhängigen Tabellen angepasst. So können Masken sehr einfach kopiert werden. Eine neue TID bekommt man durch die Wahl der nächsten Zehnerzahl, die größer als die größte vorkommende Nummer ist. Die größte vorkommende Nummer erhält man durch Ausführung des folgenden SQL-Ausdrucks mit Hilfe des Kommandos DBACCESS:

```
select max(tid) from maskeninfo;
```



Um den Austausch von Abfragen innerhalb der Hochschulen zu erleichtern ("Abfragen-Pooling" über die SuperX-Website), sollten die Masken immer im Nummernkreis xxxx0000 bis xxxx9990 liegen, wobei xxxx der von der HIS verwandten Hochschulnummer entspricht. Die Zehnerschritte ergeben sich daraus, dass die dazwischen liegenden Nummern für die Maskenfelder (Tabelle `felderinfo`) reserviert sind[20].

Wie im Abschnitt Userverwaltung beschrieben, kann die neue Maske Benutzern oder Gruppen zugänglich gemacht werden.

#### 3.1.1.8.4 Eine Maske löschen

Um eine Maske zu löschen, müssen Sie die Einträge in den oben genannten Tabellen entfernen. Dafür gibt es das Script `sx_delete_mask`

**sx_delete_mask**

| | |
|---|---|
| Aufruf: | `sx_delete_mask <TID>` |
| Aktion: | `sx_delete_mask` löscht alle Metadaten aus den Tabellen `maskeninfo`, `felderinfo`, `masken_felder_bez`, `sachgeb_maske_bez` und `maske_system_bez` zur Maske mit tid = <TID>. |

### 3.1.1.9 Änderungen an einer Maske vornehmen

1. Selektieren der Metadaten der betreffenden Maske: `sx_select_mask <TID>`

[20] Aus historischen Gründen liegen die Nummern aus Karlsruhe im Bereich 0-9990, aus Duisburg im Bereich 10000-19990.

2. Editieren der fünf Metadaten-Dateien ,,<TID>_..."
3. Abspeichern der neuen Metadaten: `sx_insert_mask <TID>`

### 3.1.1.10Ausführen von JasperReports

Neben der Ausführung im Browser gibt es eine "Kommandozeilenversion" des Aufrufs: sx_jasper.x

| Aufruf: | `sx_jasper.x -JRXML:<JRXML-Datei> -XML:<Datei mit XML-Datenquelle> -db_properties:Pfad_zur_db.properties -IGNORE_PAGINATION:<true oder false> und optional -JASPER:<Jasper-Datei> -JRPRINT:<Jrprint-Datei> -OUT:<Ausgabedatei>` |
|---|---|
| Aktion: | `sx_jasper.x` führt einen JasperReports-Task aus. Die Datenquelle kann entweder xml sein (Parameter -XML), oder eine Datenbankverbindung in der Datei db.properties. Das Ergebnis wird in eine Datei <Ausgabedatei> ausgegeben. Wenn keine Ausgabedatei angegeben wird, wird der jrxml-Dateiname verwendet, und eine PDF-Ausgabe erzeugt. |

### 3.1.1.11XSL-Transformation

Neben der Ausführung im Browser gibt es eine "Kommandozeilenversion" des Aufrufs: `sx_transform.x`

| Aufruf: | `sx_transform.x -IN:<xml-Datei> -XSL:<xsl-Datei> -OUT:<Ausgabedatei> -method:<Ausgabeformat (text,xml,html,pdf,rtf)>(optional) -param:<Parameter>(optional)` |
|---|---|
| Aktion: | `sx_transform.x` transformiert eine XMl-Datei via XSL. Wenn pdf als Ausgabeformat angegeben ist, dann wird eine PDF-Datei erzeugt. |

## 3.1.2Administration mit Abfragen im XML-Frontend

Die Masken des XMl-Frontends erscheinen bei der Anmeldung von Benutzern, die Administratorrechte haben (z.B. voreingestellte User **superx** und **admin**).

aufruft. Nach der Anmeldung erscheint die folgender Themenbaum:

**http://rechnername:8080/superx/xml/**

| | |
|---|---|
| **im XML-Frontend** | SuperX |
| | **Administration** |
| **Benutzerverwaltung** | Benutzer |
| | Gruppe suchen |
| | Institutionsrechte |
| | Nutzungsprotokolle (intern) |
| | User einrichten |
| | User löschen |
| | User suchen |
| **Organigramm** | **Institutionen suchen** |
| **Masken und Felder** | Masken |
| | Beschriftungen suchen |
| | Felder |
| | Feld erzeugen |
| | Feld kopieren |
| | Feld löschen |
| | Feld suchen |
| | Maske kopieren |
| | Maske löschen |
| | Maske suchen |
| **Sichten** | Sicht suchen |
| | Stylesheet suchen |
| | Themenbaum-Eintrag suchen |
| **Tabellen allgemein** | **Tabelle suchen** |

Nach Anklicken eines Unterpunkts (wie Institution suchen) erscheint auf der rechten Seite ein Dialog zur Suche des jeweiligen Eintrags.

### 3.1.2.1Das Organigramm bearbeiten

Meist wird das Organigramm aus anderen HIS-Systemen gefüllt, z.B. HISCOB. Wenn die Hochschule das Organigramm allerdigns selbst pflegt, gibt es die Möglichkeit, die Einsträge in einem einfachen Browser-Formular zu bearbeiten. Wenn man den Punkt **Institution suchen** anklickt und das Formular ab-schickt, erscheint z.B. folgendes Bild:

Ohne Einschränkung werden alle Institutionen im Organigramm angezeigt.
Mit dem rechten Button "Bearbeiten" gelangen Sie in eine Bearbeitungsmaske.

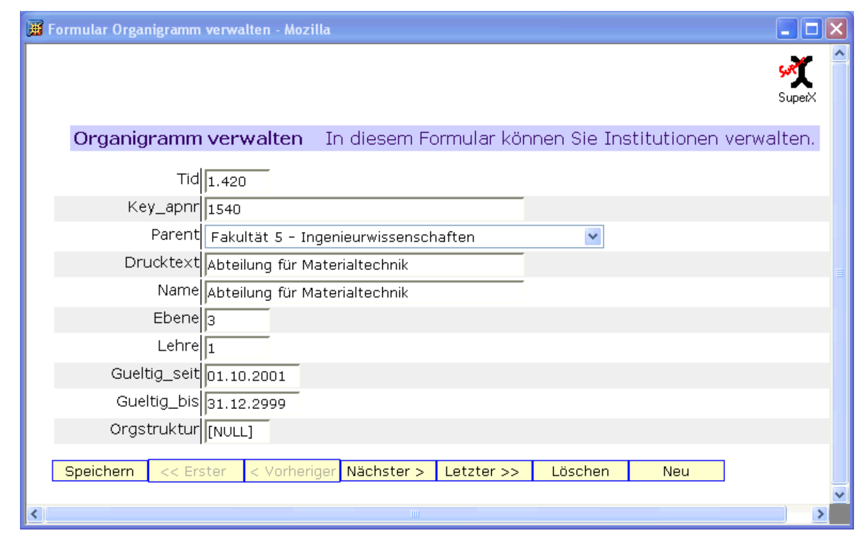
Export: Druckversion

**Institution suchen**

Stand: 01.01.2003

| Nummer | Name (kurz) | Name | Übergeordnet | Lehre | Art | Bearbeiten |
|---|---|---|---|---|---|---|
| 1510 | Abteilung für Elektrotechnik u | Abteilung für Elektrotechnik u.Informationstechnik | 1500 | 1 | | |
| 1530 | Abteilung für Informatik, Info | Abteilung für Informatik, Informations-/Medientech | 1500 | 1 | | |
| 1520 | Abteilung für Maschinenbau | Abteilung für Maschinenbau | 1500 | 1 | | |
| 1540 | Abteilung für Materialtechnik | Abteilung für Materialtechnik | 1500 | 1 | | |
| 2200 | AkZent (Akad.Beratungszentrum) | AkZent (Akad.Beratungszentrum) | 2 | 0 | | |
| 420 | Audiovisuelles Medienzentrum | Audiovisuelles Medienzentrum | 4 | 0 | | |
| 420 | Audiovisuelles Medienzentrum | Audiovisuelles Medienzentrum | 100000011 | 0 | | |
| 6440 | Beauftragte und Personalvertre | Beauftragte und Personalvertretungen | 6 | 0 | | |

Die Bearbeitungsmaske ermöglicht die Änderung der Bezeichnung (Drucktext wird normalerweise nicht angezeigt) und der übergeordneten Institution ("Parent") sowie der Gültigkeit.


Formular Organigramm verwalten - Mozilla

Organigramm verwalten   In diesem Formular können Sie Institutionen verwalten.

| Feld | Wert |
|---|---|
| Tid | 1.420 |
| Key_apnr | 1540 |
| Parent | Fakultät 5 - Ingenieurwissenschaften |
| Drucktext | Abteilung für Materialtechnik |
| Name | Abteilung für Materialtechnik |
| Ebene | 3 |
| Lehre | 1 |
| Gueltig_seit | 01.10.2001 |
| Gueltig_bis | 31.12.2999 |
| Orgstruktur | [NULL] |

Speichern | << Erster | < Vorheriger | Nächster > | Letzter >> | Löschen | Neu

Hier kann man nun den Namen, Drucktext, die Key-Apnr, Ebene, das Lehrekennzeichen, ggfs. Kennzeichen Orgstruktur und den Gültigkeitszeitraum bearbeiten.

Wenn man den Button *Neu* anklickt, erscheint der gleiche Dialog, bei dem man den Namen, key-apnr etc. der neuen Organisationseinheit eingeben kann.

Anklicken des *löschen*-Buttons entfernt eine Organisationseinheit aus dem Organigramm.

Wenn eine Organisationseinheit verschoben werden soll, z.B. Philosophie von Fachbereich 1 nach Fachbereich 6, geht dies über die Zuweisung des "Eltern"-Elements.

Wenn Sie alle Änderungen gemacht haben, können Sie diese durch Anklicken des *speichern*-Buttons in die Datenbank übernehmen.

### 3.1.2.2Den Themenbaum bearbeiten

Wenn man den Punkt **Themenbaum-Eintrag suchen** anklickt und das Formular abschickt, erscheint z.B. folgendes Bild:

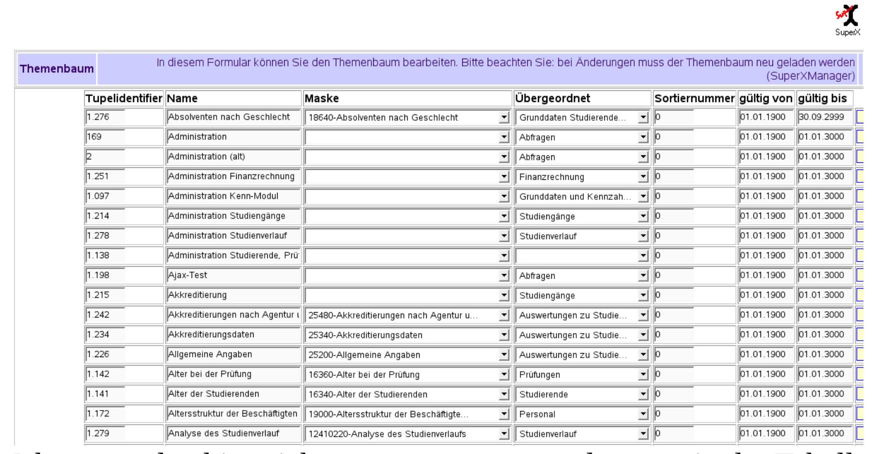
Es erscheint eine Liste mit Einträgen im Themenbaum. Sie können jeden Eintrag bearbeiten.
Einträge, die mit Masken verknüpft sind, können direkt zur Masken-Bearbeitung verlinken.

## Themenbaum-Eintrag suchen

Stand: 01.01.2003

| Name | Maske | Übergeordn. Knoten | Bearbeiten | Maske bearbeiten |
|---|---|---|---|---|
| Abfragen | | | ✎ | |
| Administration | | Abfragen | ✎ | |
| Benutzer | | Administration | ✎ | |
| Benutzer im Detail | Benutzer im Detail | Benutzer | ✎ | ✎ |
| Benutzer suchen | User suchen | Benutzer | ✎ | ✎ |
| Benutzer von SuperX | Benutzer von SuperX | Benutzer | ✎ | ✎ |
| Benutzerdaten ändern | Benutzerdaten ändern | Benutzer | ✎ | ✎ |
| Beschriftung suchen | Beschriftungen suchen | Masken | ✎ | ✎ |
| Feld erzeugen | Feld erzeugen | Felder | ✎ | ✎ |
| Feld kopieren | Feld kopieren | Felder | ✎ | ✎ |
| Feld löschen | Feld löschen | Felder | ✎ | ✎ |

Das folgende Bild zeigt die Bearbeitungsmaske. Es können Bezeichnungstexte und übergeordnete Elemente geändert werden. Beachten Sie, dass nach jeder Änderung in der jeweiligen Spalte rechts auf "Speichern" geklickt werden muss.


SuperX

Themenbaum — In diesem Formular können Sie den Themenbaum bearbeiten. Bitte beachten Sie: bei Änderungen muss der Themenbaum neu geladen werden (SuperXManager)

| Tupelidentifier | Name | Maske | Übergeordnet | Sortiernummer | gültig von | gültig bis |
|---|---|---|---|---|---|---|
| 1.276 | Absolventen nach Geschlecht | 18640-Absolventen nach Geschlecht | Grunddaten Studierende... | 0 | 01.01.1900 | 30.09.2999 |
| 169 | Administration | | Abfragen | 0 | 01.01.1900 | 01.01.3000 |
| 2 | Administration (alt) | | Abfragen | 0 | 01.01.1900 | 01.01.3000 |
| 1.251 | Administration Finanzrechnung | | Finanzrechnung | 0 | 01.01.1900 | 01.01.3000 |
| 1.097 | Administration Kenn-Modul | | Grunddaten und Kennzah... | 0 | 01.01.1900 | 01.01.3000 |
| 1.214 | Administration Studiengänge | | Studiengänge | 0 | 01.01.1900 | 01.01.3000 |
| 1.278 | Administration Studienverlauf | | Studienverlauf | 0 | 01.01.1900 | 01.01.3000 |
| 1.138 | Administration Studierende, Prü | | | 0 | 01.01.1900 | 01.01.3000 |
| 1.198 | Ajax-Test | | Abfragen | 0 | 01.01.1900 | 01.01.3000 |
| 1.215 | Akkreditierung | | Studiengänge | 0 | 01.01.1900 | 01.01.3000 |
| 1.242 | Akkreditierungen nach Agentur u | 25480-Akkreditierungen nach Agentur u... | Auswertungen zu Studie... | 0 | 01.01.1900 | 01.01.3000 |
| 1.234 | Akkreditierungsdaten | 25340-Akkreditierungsdaten | Auswertungen zu Studie... | 0 | 01.01.1900 | 01.01.3000 |
| 1.226 | Allgemeine Angaben | 25200-Allgemeine Angaben | Auswertungen zu Studie... | 0 | 01.01.1900 | 01.01.3000 |
| 1.142 | Alter bei der Prüfung | 16360-Alter bei der Prüfung | Prüfungen | 0 | 01.01.1900 | 01.01.3000 |
| 1.141 | Alter der Studierenden | 16340-Alter der Studierenden | Studierende | 0 | 01.01.1900 | 01.01.3000 |
| 1.172 | Altersstruktur der Beschäftigten | 19000-Altersstruktur der Beschäftigte... | Personal | 0 | 01.01.1900 | 01.01.3000 |
| 1.279 | Analyse des Studienverlauf | 12410220-Analyse des Studienverlaufs | Studienverlauf | 0 | 01.01.1900 | 01.01.3000 |

Die Bezeichnungen von Maksen werden hier nicht vorgenommen, sondern nur in der Tabelle `maskeninfo`.

Ein Eintrag kann in der jeweiligen Zeile durch Anklicken von *löschen* entfernt werden.

Wenn Sie eine neue Kategorie wie Administration, Studierende oder Haushalt oder neue Masken einhängen wollen, wählen Sie unten *Neu*.

Neu im Kernmodul3.5rc2 ist die Spalte . Diese ermöglicht eine andere als die alphabetische Sortierung die der Standard ist. Sie können Sie mittels Formular oder auch direkt in der Datenbank bearbeiten.

Ein Beispiel für eine nicht-alphabetische Sortierung

| **Themenbaumknoten** | **sortnr** |
|---|---|
| Personal, Stellen | 1000 |
| Studierende, Prüfungen | 2000 |
| Finanzrechnung | 3000 |
| Kostenrechnung | 4000 |

Innerhalb einzelner Knoten wird wieder alphabetisch sortiert. Wenn Sie aber z.B. Abfragen unter Kostenrechnung anders sortieren möchten, könnten Sie sortnummern von 4001 bis 4999 nutzen.

(Intern wird zuerst nach sortnummer und dann nach der Bezeichnung sortiert, wobei die Hierarchie im Baum aber bewahrt bleibt.)

### 3.1.2.3Userverwaltung

In SuperX lassen sich User- und Gruppenrechte komfortabel durch das XML-Frontend einrichten. Ausführliche Informationen zu den Tabellen und Relationen finden Sie im Kapitel Userverwaltung.

#### 3.1.2.3.1Einzelne Benutzer löschen, neu anlegen und Stammdaten ändern

Wenn Sie im Bereich Administration den Bereich *Benutzer* wählen, sehen Sie folgende Oberfläche:

Themenbaum-Menü zur Userverwaltung

**Administration**

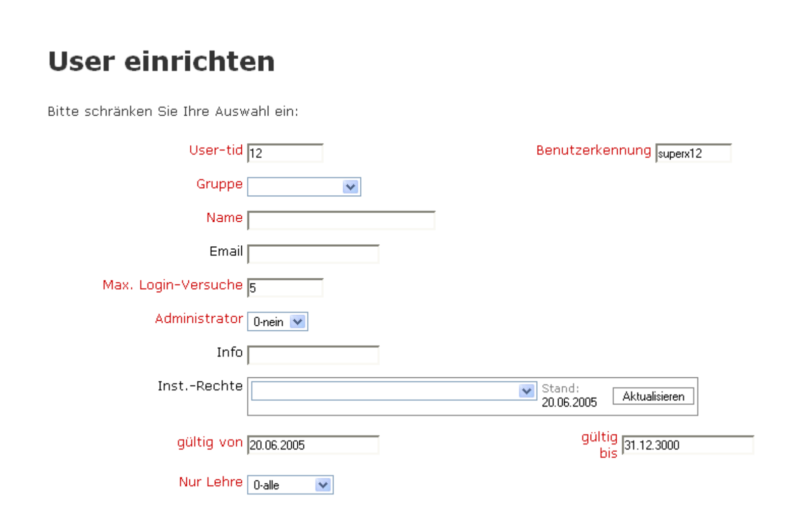
Benutzer

- Benutzer im Detail
- Gruppe einrichten
- Gruppe löschen
- Gruppe suchen
- User einrichten
- User löschen
- User suchen

##### *3.1.2.3.1.1Neuer Benutzer*

Wenn Sie einen neuen Benutzer einrichten wollen, klicken Sie auf *User einrichten*. Anschließend werden Sie nach Angaben zur Kennung für den neuen Benutzer gefragt:

Die User-tid wird automatisch hochgezählt. Die Benutzerkennung ist der Login-Text, und eine Gruppe kann ausgewählt werden. Der Name der Person muss angegeben werden. Wenn Sie das Feld "Inst-Rechte" leer lassen, hat der User Rechte auf alle Institutionen.




Die Gültigkeit kann ebenfalls eingeschränkt werden. Das Klappmenü "nur Lehre" wird bei den Orgranigramm-Rechten ausgewertet (obsolet mit 3.0).

Klicken Sie zum Abschluss auf "Abschicken". Der Benutzer wird dann mit dem verschlüsselten Passwort angelegt.

#### *3.1.2.3.1.2Benutzer löschen*

Wenn Sie einen Benutzer löschen wollen, wählen Sie im Themenbaum die Abfrage "User löschen" und dort die Kennung in der Combobox.

# User löschen

Bitte schränken Sie Ihre Auswahl ein:

Bestätigen Sie Ihre Auswahl einmal, indem Sie die Kennung eintippen.

#### *3.1.2.3.1.3Benutzer bearbeiten*

Im Formular erhalten Sie je nach Einschränkung eine Liste mit Benutzern.

Wir schränken z.B. ein auf die Gruppe Administratoren.

Es erscheinen zwei User, die voreingestellten Administratoren. Sie können sich Details zur Person ansehen, oder die Person bearbeiten.

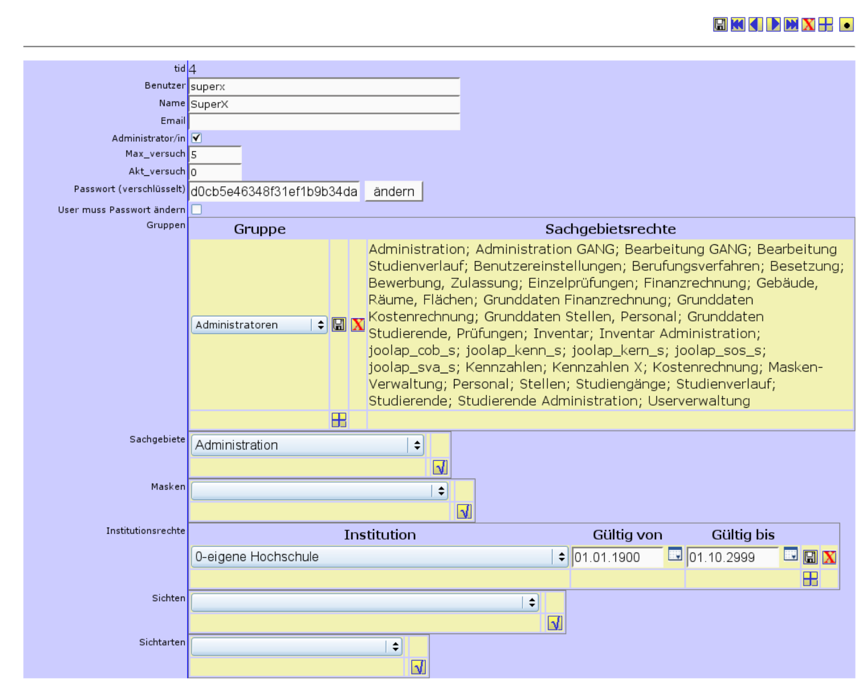
## User suchen

Gruppe: **Administratoren** ; Stand: 01.01.2003

| Kennung | Name | Email | Ansehen | Bearbeiten |
|---|---|---|---|---|
| admin | Jane Doe | | | |
| superx | John Doe | | | |

Datensatz 1 - 2 von insgesamt 2 Sätzen.

In der Bearbeitungsmaske können Sie Kennung und Namen ändern, sowie das Passwort ändern. Danach müssen Sie oben rechts auf die Diskette zum Speichern klicken.
Sie können die Gruppenzugehörigkeit zuordnen, und sie können dem User Rechte auf einzelne Sachgebiete, Masken, Institutionen, Sichten und ganze Sichtarten geben.




Bei den Gruppen, Sachgebiete, Masken, Institutionen, Sichten und Sichtarten handelt es sich um Unterformulare, d.h. Sie brauchen Änderungen hier nicht mit dem Diskettensymbol oben rechts zu speichern.

#### *3.1.2.3.1.4Zum Häkchen Administrator/in*

Das Häkchen Administrator/in hat keinen Effekt auf die sichtbaren Sachgebiete und Masken im Themenbaum. Es bewirkt folgendes:

- Lese- und Schreibzugriff für alle DBFORMS
- Leserecht für alle Sichten
- Ausführungsrecht für Administrationsmasken im Kernmodul, z.B. User löschen
- Recht, im SuperXManager den Cache zu leeren und die SQL-Protokollierung anzuzeigen
- Recht, JasperReports-Templates zu generieren (erst ab Kernmodul 4.0rc2, oder HISinOne 2.0)
- Leserechte für alle Bäume und Würfel in Joolap

> Für die Gruppen Administratoren und Bearbeiter musste bei Einsatz des SuperX-Kernmoduls 3.0 immer das Häkchen bei "Administration" gesetzt werden, denn nur diese Personen durften im Kernmodul 3.0 die DBForms überhaupt nutzen. Im Kernmodul 3.5 oder höher ist dies nicht mehr notwendig, da reicht es aus wenn die User der Gruppe "Administratoren" (Name darf nicht geändert werden) angehören, bzw. im GANG-Modul den Gruppen **GANG Bearbeiter** oder **GANG Administratoren**.

### 3.1.2.3.2Gruppen anlegen, löschen und Stammdaten verwaltung

Im Bereich *Gruppenverwaltung* erhält man die Möglichkeit, Gruppen zu löschen, neue Gruppen anzulegen und Stammdaten zu ändern.

### *3.1.2.3.2.1Neue Gruppe anlegen*

Wenn man eine neue Gruppe einrichten will, wählt man im Menü `Administration->Benutzer->Gruppe einrichten`.

Sie können eine Nummer vergeben, den Namen festlegen (darf nicht bereits existieren), die zugehörigen Mitglieder auswählen, und Rechte für Sachgebiete und Masken vergeben. In den Listen sind auch mehrere Einträge auswählbar, jeweils mit der Taste "Strg" und einem Mausklick (beim Mac die Apfel-Taste).
Mit **Abschicken** werden die Tabellen gefüllt.

**Gruppe einrichten**

Bitte schränken Sie Ihre Auswahl ein:

Gruppen-tid: 3

Name: Testgruppe

Mitglieder: Administrator, SuperX, Testuser

Sachgebiets-Rechte: Administration, Institutionsverwaltung, Kostenrechnung, Masken-Verwaltung, Personal, Stellen, Studierende

Masken-Rechte: 16020-Studierende und Studienanfänger nach Geschlecht, 16040-Studierende nach Fach, Abschluss und Geschlecht, 16060-Prüfungen nach Fachsemestern, 16100-Prüfungen nach Geschlecht + Staatsang. (Zeitreihe), 16120-Fachstudiendauer (Zeitreihe), 16140-Note nach Geschlecht + Staatsang. (Zeitreihe), 16160-Einschreibungen (Zeitreihe)

Abschicken

Sie können auch direkt im Tabellenformular arbeiten, im Menü "Tabelle suchen"-> Tabelle `groupinfo`.

Hier können Sie Gruppenbezeichnungen ändern, Gruppen löschen (Vorsicht!) und neu anlegen.
Wenn Sie neue Gruppen anlegen, müssen Sie die Nummern (`tid`) selbst festlegen, die Datenbank zählt nicht hoch.

### *3.1.2.3.2.2Gruppe bearbeiten*

Im Formular erhalten Sie je nach Einschränkung eine Liste mit Gruppen.

Wir schränken z.B. ein auf die Gruppe(n) des Users **superx**.

SuperX

20.04.2005 hilfe | über

## Gruppe suchen

Bitte schränken sie Ihre Auswahl ein:

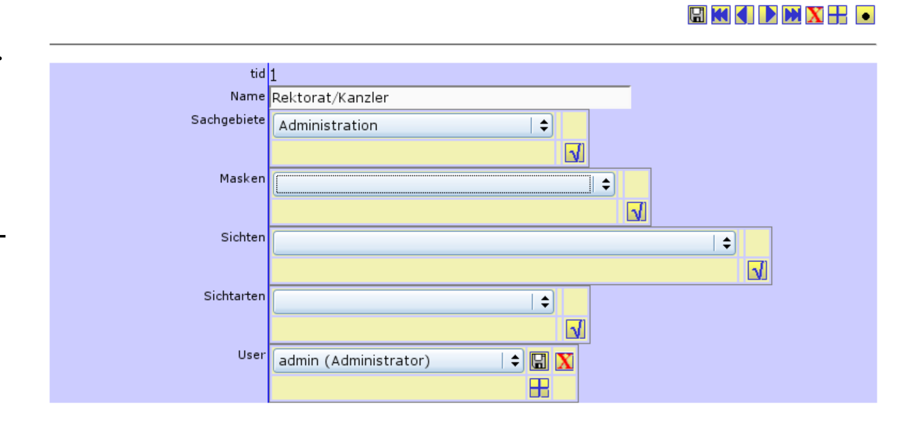
Benutzer superx - John Doe

Name

Abschicken Zurücksetzen

Es erscheint ein Eintrag mit der Gruppe Administratoren, die zwei User enthält. Sie können die Gruppe bearbeiten.

In der Bearbeitungsmaske können Sie den Namen der Gruppe ändern. Danach müssen Sie unten "Speichern" anklicken.
Sie können die Gruppenzugehörigkeit zuordnen, und sie können der Gruppe Rechte auf einzelne Masken, Sichten und Sichtarten geben. Außerdme können Sie der Gruppe weitere User zufügen / enfernen.




Bei den Gruppen, Masken, Sichten, Sichtarten und Usern handelt es sich um Unterformulare, d.h. Sie brauchen Änderungen hier nicht mit dem Diskettensymbol oben rechts zu speichern.

***3.1.2.3.2.3Gruppe löschen***

Im XML-Frontend im Menü Administration -> Benutzer -> Gruppe löschen können Sie eine Gruppe löschen und die jeweiligen Rechte für Sachgebiete und Masken entfernen. Auch die Zuordnungen von Usern zur Gruppe (nicht aber die User selbst) werden gelöscht.

Sie müssen lediglich den Namen der Gruppe auswählen und einmal zur Sicherheit bestätigen, indem Sie den Namen eintippen.
Mit **Abschicken** werden die Einträge entfernt.

Gruppe löschen

Bitte schränken Sie Ihre Auswahl ein:

zu löschende Gruppe Testgruppe

Name (bestätigen) Testgruppe

Abschicken

## 3.1.3Rechte für DBFORMS

Die DBFORMS dienen der Dateneingabe in SuperX, z.B. für die Konfiguration. Alle DBFORMS lassen sich direkt aus einer Maske aufrufen und sind daher nur für die Personen sichtbar, die auch das jeweilige Maskenrecht haben.

Wenn die User das Recht auf das Sachgebiet des jeweiligen DBFORMS haben (Tabelle `sachgeb_dbform_bez`), bekommen sie Leserecht, sofern das DBFORM "stand-alone" arbeitet, d.h. nicht mit Pflichtparameter aus einer Maske aufgerufen wird, sondern direkt über die Maske "Tabelle suchen".

Wenn die User einer Gruppe zugeordnet sind, die Rechte auf Sachgebiete mit dem Namen "Administration*" oder "Bearbeitung*" haben, bzw. wenn sie als Einzeluser Recht auf ein solches Sachgebiet haben, bzw. wenn sie das Admin-Häkchen haben, bekommen sie auch Schreibrecht (Daten einfügen, löschen, neu erzeugen). Beim Modul GANG sind das z.B. die Sachgebiete "Administration GANG" und "Bearbeitung GANG".

## 3.1.4Hochschulspezifische Filter anlegen

In fast jedem SuperX-Modul gibt es die Möglichkeit, hochschuleigene Filter anzulegen. Die Maskenfelder dazu lauten "Filter Studierende", "Filter Personal" etc. Hier ein Beispiel:

Der Button "Filter Studierende":

Hinter dem Namen des Filters verbirgt sich eine SQL-where-Bedingung. Die Bedingung wird vor dem Hintergrund der jeweiligen Hilfstabelle formuliert, hier z.B. die Hilfstabelle "Studierende" im SOS-Modul. Die zugehörige Tabelle finden Sie auf der Seite der Datenbankbeschreibung des Moduls, hier z.B.

http://www.studio-fuer-textdesign.de/superx/doku/sos_modul/sos.html

Dort schauen Sie rechts in der Spalte "Hilfstabellen", welche Tabellen es gibt. Die gesuchte Tabelle lautet `sos_stg_aggr`

http://www.studio-fuer-textdesign.de/superx/doku/sos_modul/sos.html#tab_sos_stg_aggr

Wenn Sie z.B. einen Filter "nur weibliche Studierende"erzeugen wollen, wählen Sie zunächst im Maskenfeld "Geschlecht den gewünschten Wert:

Geschlecht weiblich

Klicken Sie auf den Button "Schlüssel anzeigen" . Danach sehen Sie den Wert des Schlüssels:

Geschlecht 2 weiblich

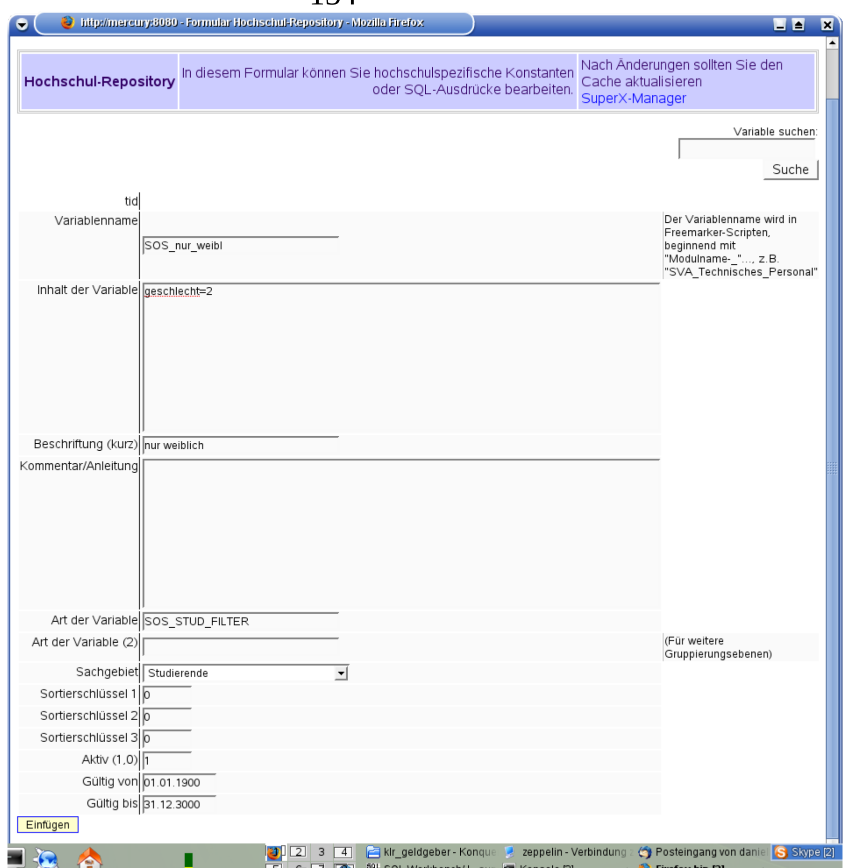
Der Wert für weiblich ist "2". Dann wäre die Bedingung:

| **Filter "nur weiblich"** | geschlecht=2 |
|---|---|

Den Inhalt des Filters können Sie in der Tabelle "Hochschul-Repository" einpflegen: Gehen Sie im Browser in das Menü "Administration->Tabelle suchen", geben Sie beim Stichwort "repository" ein, und klicken Sie auf "Suchen". Sie erhalten einen Datensatz :

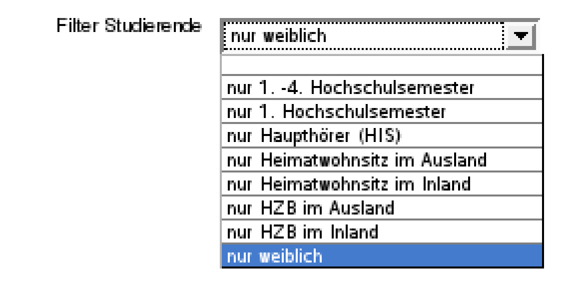
Hier klicken Sie auf "Bearbeiten". Sie erhalten ein Datenbankformular, wo Sie rechts nach Variablen suchen können. Am besten suchen Sie eine Variable, die es schon gibt, indem Sie den Modulnamen eingeben, z.B. hier "SOS". Sie erhalten verschiedene Beispielfilter, allen ist gemeinsam, daß sie im Feld "Art der Variable" den Wert "SOS_STUD_FILTER" haben. Wenn Sie einen neuen Filter eingeben wollen, gehen Sie unten auf den Button "Neu". Dann geben Sie die Werte ein:

Vergeben Sie einen eindeutigen Namen, z.B. "SOS_nur_weib", im Feld "Inhalt" schreiben sie die where Bedingung, und die Beschriftung erscheint dann in der Maske.
Wichtig ist der Wert bei "Art der Variable", das Sachgebiet, der Schalter "Aktiv", und die Gültigkeit.
Wenn Sie das Formular mit "Einfügen" abschicken, erscheint wieder die komplette Liste, der Datensatz ist am Ende angefügt.




Danach gehen Sie im Manager auf Cache leeren, und öffnen eine Studierenden Maske erneut:

Der Filter ist nun sichtbar und nutzbar - in allen Masken zu Studierenden.

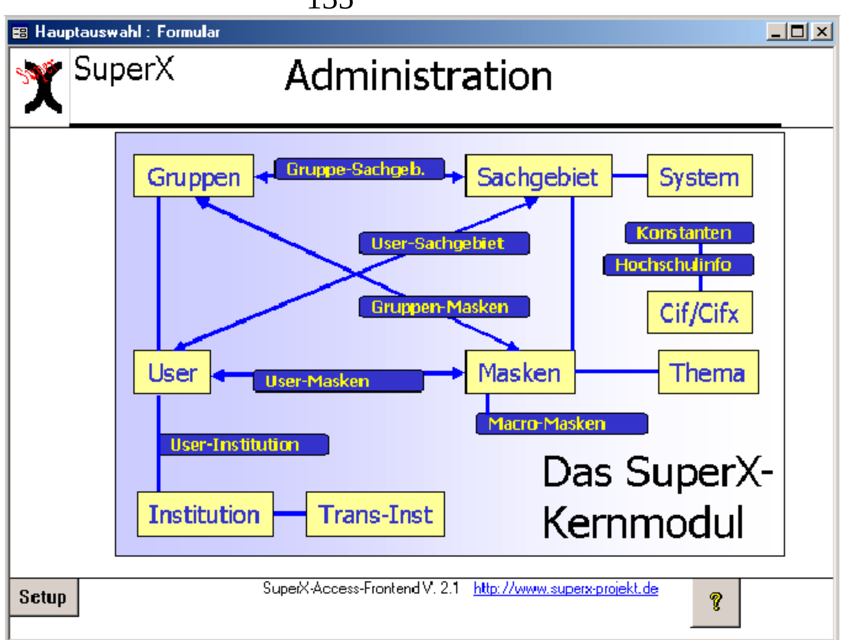



Sie können auch komplexere Filter einbauen, z.B. "nur Haupthörer, ohne 1. Hochschulsem., ausl. Staatsangehörigkeit", indem Sie die where-Bedingungen mit "and" verknüpfen. Achten Sie bei der Syntax darauf, dass die SQL-Syntax nicht zerstört wird. Bei alphanumerischen Feldern müssen Sie z.B. immer ein einfaches Hochkomma um die Werte setzen.

### 3.1.5Das Access-Frontend

Die Access-Datenbank enthält die Tabellen des Kernmoduls als Verknüpfungen und ermöglicht so ein leichtes Administrieren der Datenbank. Die Installation ist in der Installationsanleitung für ODBC-Quellen beschrieben. Die folgende Abbildung zeigt das Hauptmenü:

Das Frontend eignet sich zur Verwaltung von Usern, Gruppen, Sachgebieten und Masken sowie deren relationalen Verküpfungen (blaue Kästchen). Darüberhinaus sind Formulare für das Systeminfo, den Themenbaum und das Organigramm vorgesehen.




Probleme mit der Bedienung von Access gibt es immer dann, wenn Tabellen keine Primärschlüssel haben oder wenn die Felder mit den Primärschlüsseln nicht gefüllt sind. Mit der Version 2.1 erhalten alle Tabellen in SuperX (außer Datentabellen und Hilfstabellen, weil diese normal nicht manuell bearbeitet werden) Primärschlüssel. Wenn es dennoch Probleme gibt, empfehlen wir die Java-basierte SQLWorkbench.

Das Access-Frontend ist insbesondere für die Änderung von Masken und Feldern gut geeignet.

### 3.1.6Weitere Tools

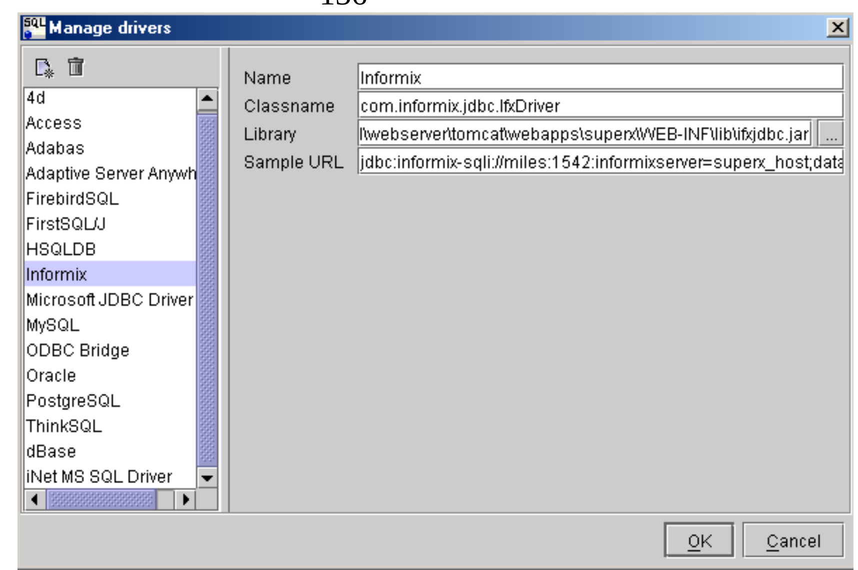
Durch die odbc- und jdbc-Treiber können beliebige Datenbankfrontends eingesetzt werden. Gute Erfahrungen gerade mit Tabellen ohne Primärschlüssel haben wir mit der SQLWorkbench von Thomas Kellerer gemacht. Exemplarisch für andere jdbc-Clients haben wir dieses Programm näher beschrieben.

#### 3.1.6.1SQLWorkbench

Die SQLWorkbench von Thomas Kellerer arbeitet mit dem jdbc-Treiber jeweils von Postgres oder Informix. Beim ersten Aufruf der Workbench können Sie Profile für Treiber und Datenbanken eingeben. Musterprofile für viele gängige Datenbanksysteme liegen vor. Leider ist der Informix-Treiber nicht dabei, deshalb muss dieser "von Hand" registriert werden. Gehen Sie dazu über File->Connect in das Feld "Manage Drivers". Dort können Sie einen Namen vergeben und die jdbc-Parameter übertragen. Die folgende Abbildung zeigt ein Beispiel:

Der Dialog zur Einrichtung von Datenbanktreibern am Beispiel Informix.
Die Parameter entsprechen denen, die Sie für das SuperX-Servlet in db.properties definieren.

Der Informix-Treiber `ifxjdbc.jar` muss lokal gespeichert sein.



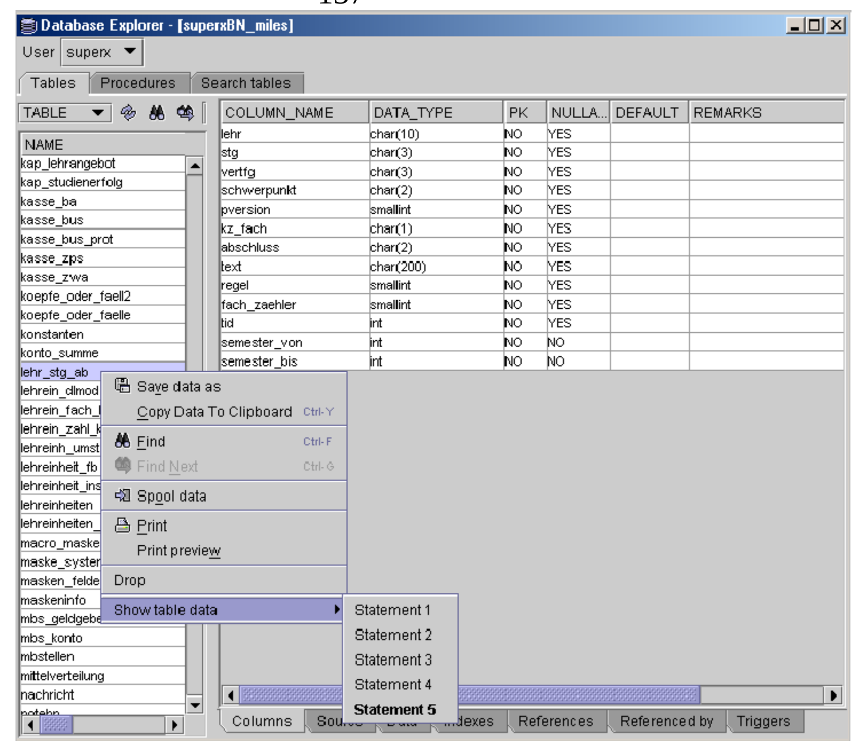

Im Dialog File -> Connect können Sie dann eine Datenquelle eintragen, und die Verbindungsparameter vervollständigen (Username, Passwort). Autocommit sollten Sie immer einschalten.

Die SQLWorkbench ist ein hervorragendes Administrations- und Entwicklungswerkzeug, daher haben wir die Version 94 in das SuperX-Clientpaket 30final integriert. Darin sind die Profile und Treibereinstellungen für Informix, Postgres und Joolap bereits voreingestellt, Sie müssen lediglich Servernamen und Ports ändern.

Interessant ist der Datenbank-Explorer (Tools -> Database Explorer), der es ermöglicht, die Datenbank nach Tabellen / Prozeduren etc. zu durchsuchen. Wenn eine Tabelle ausgewählt ist, kann sie auch über die Registerkarte "Data" editiert werden. Achten Sie darauf, dass Sie das Feld **Max. Rows** auf einen sinnvollen Wert setzen, z.B. 2000. Die SQLWorkbench ist gerade für die Arbeit mit Tabellen ohne Primärschlüssel geeignet, weil jede Äderung intern als Update formuliert wird. Der Nachteil ist, dass nicht mehrere Zellen über Zwischenablage geändert / eingefügt werden können.

Sehr praktisch für die Entwicklung von SQL-Abfragen ist die Möglichkeit, zu jeder Tabelle einen select-String zu formulieren.

Markieren Sie die Tabelle im Database Explorer, und gehen Sie über das Kontextmenü auf Show table data, und wählen Sie ein Editorfenster aus. Der Select-String wird dann angezeigt.




Seit den Versionen 93 lassen sich bei Informix auch Felder vom Typ `text` anzeigen und editieren.

Das Tool bietet außerdem eine Makrofunktion, und in neueren Versionen auch ETL-Funktionen über einen "Data Pumper", was es natürlich für SuperX besonders interessant macht. Weitere Tipps und Hilfen erhalten Sie im (gelungenen, aber englischen) Benutzerhandbuch der Workbench.

## 3.2 Einen User betreuen

Jeder Benutzer von SuperX sollte ein geheimes Paßwort benutzen, welches nicht einfach erraten werden kann. Paßwörter wie Vornamen, Stellung im Beruf o.ä. dürfen unter keinen Umständen verwendet werden. Zum Ändern des Paßworts kann im Applet und im XML-Frontend ein Paßwortänderungsdialog aufgerufen werden.

### 3.2.1 Neuen User einrichten

Im Kernmodul befindet sich eine Abfrage "User einrichten", mit der Sie einen User einrichten und ggf. auf bestimmte Institutionsrechte oder Gültigkeitszeiträume einschränken können. Außerdem können Sie den User einem Sachgebiet oder einer Gruppe zuordnen.

Bearbeitungsformulare zur Benutzerverwaltung befinden sich im XML-Frontend. Ausführliche Informationen finden Sie im Kapitel UserverwaltungTabellen. Hier eine Anleitung für die direkte Änderung in der Datenbank:

1. Erstellen eines Eintrags in der Tabelle `userinfo`. Neue Tid merken.
2. Setzen des Startpassworts z.B. "anfang12". Mit dem Befehl

```
update userinfo set passwd_sha="0533a66a3e9bea16f3139bfe4f6ce50ced591dea" where tid=<Neue Tid>
```

Der User muss aufgefordert werden, sein Passwort beim ersten Start zu ändern.

3.Dem User Rechte für Institutionen geben durch Einträge in die Tabelle `user_institution`.
Ggfs. Gruppenzugehörigkeit eines Benutzers festlegen.

4.(Eintrag in Tabelle `user_group_bez` - siehe Abschnitt Userverwaltung)

5.Falls durch die Gruppenrechte noch nicht abgedeckt: Dem User Rechte für ganze Sachgebiete und/oder einzelne Abfragen geben (Einträge in die Tabellen `user_sachgeb_bez` bzw. `user_masken_bez`)

Sie können einen User zwingen, sich nach der Anmeldung ein neues Passwort zu geben. Dazu wird dem Feld  in der Tabelle  der Wert "ändern" gegeben. Es erscheint dann nach der ersten Anmeldung eine Aufforderung, das Passwort zu ändern.

### 3.2.2 Passwort vergessen

Den Befehl

```
update userinfo set passwd_sha="0533a66a3e9bea16f3139bfe4f6ce50ced591dea" where benutzer="<Kennung des Benutzers">
```

ausführen. Dadurch erhält der Benutzer wieder das Startpasswort "anfang12", was er nach erfolgreichem Anmelden wieder sofort ändern sollte. Des weiteren kann der SuperX-Adminstrator im XML-Frontend ein beliebiges Passwort für den Benutzer vergeben und die Checkbox „User muss Passwort ändern“ aktivieren, damit der er bei der nächsten Anmeldung ein neue Passwort vergeben muss.

### 3.2.3 User-Rechte ändern

- Rechte für Institutionen gibt oder entfernt man durch Hinzufügen/Löschen von Einträgen in der Tabelle `user_institution`
- Gruppenzugehörigkeit wird über die Tabelle `user_group_bez` definiert, ggfs. dort Einträge hinzufügen oder löschen
- individuelle Rechte für Sachgebiete und/oder Masken über die Tabellen `user_sachgeb_bez` bzw. `user_masken_bez` anpassen

Die Stammdaten (Name, email, etc) befinden sich in der Tabelle `userinfo`.

### 3.2.4 User löschen

Es gibt im Kernmodul eine Abfrage "User löschen" (im Themenbaum unter Administration -> Benutzerverwaltung). Wenn Sie den User "von Hand" löschen wollen:

Die tid des Benutzers aus der Tabelle `userinfo` heraussuchen.

Folgende Befehle ausführen.

```
delete from user_masken_bez where userinfo_id=<tid des Users>;
delete from user_sachgeb_bez where userinfo_id=<tid des Users>;
delete from user_institution_bez where userinfo=<tid des Users>;
delete from userinfo where tid=<tid des Users>;
```

## 3.3Einstellungen zur Passwortsicherheit

Bei der Installation des SuperX-Kernmoduls werden in die Tabelle `konstanten` vier Einträge zur Einstellung der Passwortsicherheit gemacht.

| Name der Konstante | Kommentar | default-Wert |
|---|---|---|
| Passwortgültigkeit (Tage) | Gibt an, für wie viele Tage das Passwort gültig sein soll. Sobald die Gültigkeit abgelaufen ist, muss der Anwender bei der nächsten Anmeldung ein neues Passwort vergeben. | 180 |
| Passwort Groß- u. Kleinb. | Müssen Groß- und Kleinbuchstaben im Passwort vorkommen (0=nein, 1=ja) | 1 |
| Passwort erfordert Ziffer | Müssen Ziffern im Passwort vorkommen (0=nein, 1=ja) | 1 |
| Passwortlänge (Minimum) | Geben Sie hier die minimale Passwortlänge an | 8 |

Beim Upgrade einer älteren Kernmodul-Installation (vor Kernmodul 3.0rc7) sind folgende Vorbelegungen aktiv:

| Name der Konstante | default-Wert |
|---|---|
| Passwortgültigkeit (Tage) | 360 |
| Passwort Groß- u. Kleinb. | 0 |
| Passwort erfordert Ziffer | 0 |
| Passwortlänge (Minimum) | 6 |

Die Zentrale Datenschutzstelle der baden-württembergischen Universitäten (Zendas) macht folgende Empfehlung:

| | |
|---|---|
| Passwortgültigkeit (Tage) | 90-180 |
| Passwort Groß- u. Kleinb. | 1 |
| Passwort erfordert Ziffer | 1 |
| Passwortlänge (Minimum) | 8 |

Um die Konstanten zu ändern, gehen Sie als Administrator in die Anwendung, gehen Sie in das Menü "Administration"->"Tabelle suchen" und suchen Sie die Tabelle "konstanten". In der Zeile klicken Sie auf den "Bearbeiten"-Button, und suchen dort die Konstante Passwortgültigkeit (Tage) etc.
Um kurzfristig die Gültigkeit aller User auf unendlich zu setzen (z.B. bei Testbetrieb), müssen sie in der Datenbank folgenden Update ausführen: `update user_pw set pw_gueltig_bis=date_val('01.01.3000');`
Der SuperX-Administrator kann erzwingen, dass der Benutzer sein Passwort ändern muss, indem er im XML-Frontend den entsprechenden User bearbeitet und bei "User muss Passwort ändern" ein Häkchen setzt. Neue User werden in der Maske "User einrichten" defaultmäßig so eingestellt, dass sie ihr Passwort nach der ersten Anmeldung ändern müssen.

## 3.4Eine Gruppe betreuen

Sie können in SuperX durch Einträge in den Tabellen zur Gruppenverwaltung einzelne Gruppen anlegen, mit Leserechten für Abfragen und Institutionen versehen. Die zugehörigen Tabellen werden in der Gruppenverwaltung erläutert.

### 3.4.1Neue Gruppe einrichten

Ausführliche Informationen zur Gruppenverwaltung finden Sie im Kapitel Userverwaltung.

1. Erstellen eines Eintrags in der Tabelle `groupinfo`. Neue Tid merken.
2. Der Gruppe Rechte für Sachgebiete und/oder einzelne Masken geben (Einträge in die Tabellen group`_sachgeb_bez` bzw. `group_masken_bez`)

### 3.4.2Gruppen-Rechte ändern

Gruppenrechte für Sachgebiete und einzelne Masken werden in den Tabellen `group_sachgeb_bez` bzw. `group_masken_bez` festlegt. Dort ggfs. Einträge machen oder löschen.

Der Gruppenname kann in der Tabelle `groupinfo` geändert werden.

### 3.4.3Eine Gruppe löschen

Die tid der Gruppe aus der Tabelle `groupinfo` heraussuchen.

Folgende Befehle ausführen.

```
delete from group_masken_bez where groupinfo_id=<tid der Gruppe>;
delete from group_sachgeb_bez where groupinfo_id=<tid der Gruppe>;
delete from groupinfo where tid=< tid der Gruppe>;
```

## 3.5Verwaltung und Rechtevergabe von Sichten

SuperX-Sichten sind hierarchische Zusammenstellungen von Dimensionen, z.B. von alternativen Kostenstellenhierarchien. Die Sichten können in einem eigenen Formular verwaltet werden, außerdem können die Berechtigungen für Sichten eingeschränkt werden.

Nach der Anmeldung als Administrator im XML-Frontend können Sie im Themenbaum die Abfrage **Administration** -> **Sicht suchen** wählen, und abschicken.

Im Kernmodul ist nur eine Sicht enthalten, eine alternative Hierarchie, die den Themenbaum aufbaut.

Sicht suchen

Stand: 01.01.2003

| Name | Beschreibung | Art | Bearbeiten | User- und Gruppenrechte |
|---|---|---|---|---|
| Themenbaum | | Themenbaum-Sicht | | |

Datensatz 1 - 1 von insgesamt 1 Satz.

## 3.5.1Bearbeitung von Sichten

Sie können mit Klick auf den "Bearbeiten"-Button die Sicht in einem Datenbank-Formular bearbeiten. Die folgende Abbildung zeigt das Formular.

Formular Sichten - Mozilla <2>

Sichten — In diesem Formular können Sie Sichten bearbeiten

| Feld | Wert | Beschreibung |
|---|---|---|
| tid | 11 | |
| Systeminfo | Administration | Querverweis zur Systeminfo |
| Art | Themenbaum-Sicht | |
| Type | 10 | 10 ist standardsicht, 20 ist alt.Hierarchie |
| Interner Name | memtext_themenbaum | Eindeutiger Datensatzbezeichner |
| Name | Themenbaum | Bezeichnung der Sicht, die für User angezeigt wird |
| Beschreibung | | |
| Sortiernummer | 0 | kann für Sortierungen benutzt werden |
| Quelle | <<SQL>> select name,tid,parent from themer | Angabe einer Tabelle mit key,parent,name,gueltig_seit,gueltig_bis oder Prozedur, die mit sp_ anfängt |
| Alt_hier_id | | id der alt.Hier in angegebenen Quelltabelle, null bei regulärer Hierarchie |
| Treecfgtable | | Tabelle mit Infos zu TreeView aus Cob |
| Treecfgid | | id des benutzen trees aus Cob-Tabelle trees null bei regulärer Hierachie |
| User_rechte | 1 | sollen User Rechte berücksichtigt werden, derzeit von Org.Sichten ausgewertet |
| Standbutton | 0 | soll der Stand geändert werden können |
| Attribut1 | | bei Bedarf noch Attribute der Sicht hinterlegt werden, auf die man bei Bedarf Einschränkungen fahren kann |
| Attribut2 | | bei Bedarf noch Attribute der Sicht hinterlegt werden, auf die man bei Bedarf Einschränkungen fahren kann |
| Attribut3 | [No Dat: | bei Bedarf noch Attribute der Sicht hinterlegt werden, auf die man bei Bedarf Einschränkungen fahren kann |
| Attribut4 | [No Dat: | bei Bedarf noch Attribute der Sicht hinterlegt werden, auf die man bei Bedarf Einschränkungen fahren kann |
| Max. Zeilen XML | [No Dat: | Max. Zeilen im Klappmenü |
| Gueltig_seit | Jan 1, 1900 | |
| Gueltig_bis | Jan 1, 3000 | |
| Aktiv | 1 | |

Speichern | << Erster | < Vorheriger | Nächster > | Letzter >> | Löschen | Neu

Wenn Sie Sichten für die Anwender ausblenden wollen, ist es nicht ausreichend, die Datensätze zu löschen - sie würden beim nächsten Update aus dem Quellsystem wieder eingespielt. Stattdessen sollten Sie die Sichten in der Konstante "Aktiv" ganz unten im Formular auf "0" setzen und dann im SuperXManager den Cache aktualisieren.

Wenn es pro Sichtart mehrere Sichten gibt, kann deren Sortierung im Feld "Sortiernummer" beeinflusst werden. Die erste Sicht in der Sichtart ist auch immer die Sicht, die im Browser-Client standardmäßig beim Aufruf der Maske angezeigt ist wird.

Weitere Details zur Anpassung von Sichten finden Sie im SuperX-Entwicklerhandbuch.

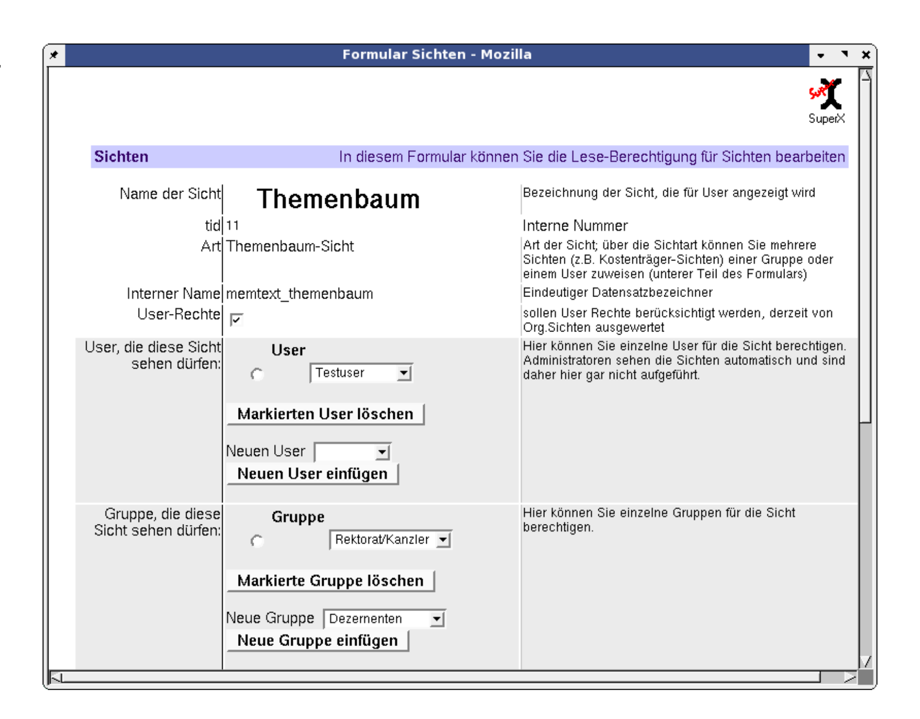
## 3.5.2Berechtigung für Sichten

Um die Berechtigung von Sichten zu vereinfachen, werden mehrere Sichten in SuperX zu "Sichtarten" zusammengefasst. Sie können Berechtigungen auf beiden Ebenen vergeben.

### 3.5.2.1User- und Gruppenrechte für Sichten

Mit Klick auf den Button "User- und Gruppenrechte" können Sie die Rechte für die Sicht / Sichtart vergeben.

Wie beid er Userverwaltung finden Sie hier einige Unterformulare: User bzw. Gruppen, die die Sicht sehen dürfen. Sie können User bzw. Gruppen hinzufügen oder entfernen.




Im unteren Teil des Formulars können Sie User- und Gruppenrechte für ganze Sichtarten festlegen.

User, die die ganze Sichtart sehen dürfen:
User
Testuser
Hier können Sie einzelne User für die Sicht berechtigen. Administratoren sehen die Sichten automatisch und sind daher hier gar nicht aufgeführt.
Markierten User löschen
Neuen User
Neuen User einfügen
Gruppen, die die ganze Sichtart sehen dürfen:
User
Rektorat/Kanzler
Hier können Sie einzelne Gruppen für die Sicht berechtigen
Markierte Gruppe löschen
Neue Gruppe
Dezernenten
Neue Gruppe einfügen

Wie bei der Userverwaltung handelt es sich um Unterformulare, d.h. Sie brauchen die Änderungen jeweils nicht zu manuell zu speichern.

#### 3.5.2.2 Sachgebiete und Sichten

Bei der Installation des jew. Moduls erhalten alle User, die Rechte auf das Sachgebiet haben, z.B. "Finzanzrechnung", auch Rechte auf alle Sichten im Bereich Finanzrechnung. Man kann diese Rechte auch nachträglich für einzelne Sichten entfernen, indem man wie folgt vorgeht:

- Im XML-Frontend anmelden als Administrator, und zur Maske "Tabelle suchen" gehen, dort die Tabelle "`sachgeb_sichtarten`" bearbeiten
- In der Tabelle den Eintrag z.B. für die Zuordnung des Sachgebiets Finanzrechnung zu FIN-Kostenstellen-Sichten löschen.
- Dann in die Tabelle `sachgeb_sichten` gehen und Bearbeiten
- Dort einen neuen Datensatz mit dem Sachgebiet "Finanzrechnung" und der regulären Sicht FIN-Kostenstellen erzeugen
- Dann einen neuen Datensatz mit dem Sachgebiet "Finanzrechnung" und z.B. einer internen Sicht "FIN-Kostenstellen intern" erzeugen

Damit haben alle User mit Recht auf das Sachgebiet Finanzrechnung automatisch auch Recht auf die beiden Sichten: reguläre Sicht und die interne Sicht FIN-Kostenstellen.

Weitere Sichten können einzelnen Usern/Gruppen dann über die Maske "Sicht suchen"->User und Gruppenrechte vergeben werden (s.o.).

Danach im Manager den  Cache leeren und neu anmelden.

#### 3.5.2.3 Kostenstellenrechte innerhalb von Sichten

##### 3.5.2.3.1 Reguläre Sicht

Oben wurde dargestellt, wie Leserechte für Sichten vergeben werden. Für eine spezielle Form von Sichten ist es darüber hinaus auch möglich, innerhalb einer Sicht bzw. Hierarchie auf einzelne Knoten einzuschränken: für Kostenstellen-Sichten (erkennbar an der Sichtart "XXX-Kostenstellen-Sicht"). Dies wollen wir an einem Beispiel verdeutlichen:

Angenommen wir haben eine "Fakultät 1 für Geisteswissenschaften". Innerhalb dieser Fakultät gibt es Lehreinheiten, und darunter Institute bzw. Professoren. Ein Auszug aus dem Beispielbaum:

•Fakultät 1 für Geisteswissenschaften

◦Lehreinheit Geschichte

■Institut für Frühgeschichte

•Prof. Meyer

◦Lehreinheit Philosophie

■Institut für Humanistische Philosophie

•Prof.'in Schulze

Der Beispielbaum bildet das Organigramm der Hochschule, in HISinOne entspricht dies der Tabelle `orgunit`. Dies ist gleichzeitig die Grundlage für die Zuweisung von Benutzerrechten. Eine Person kann einem oder mehreren "Knoten" im Organigramm zugeordnet werden. So könnte man z.B. einstellen, daß Prof.'in Schulze nur ihre eigene Kostenstelle sehen darf, nicht die übergeordneten.

Die Berechtigung gilt normalerweise systemweit, d.h. in allen Auswertungen, egal ob im Haushalts,- Personal-, Flächen,- Inventar- oder KLR-Bereich, sind diese Rechte wirksam.

Nun kann es notwendig sein, bereichsspezifisch alternative Berechtigungen zu implementieren. An dieser Stelle kommen alternative Hierarchien ins Spiel, wie im Folgenden an einem Beispiel gezeigt wird.

#### 3.5.2.3.2Rechte innerhalb von alternativen Hierarchien

Neben der Berechtigung innerhalb der regulären Sicht bzw. dem Organigramm kann es notwendig sein, auch Rechte innerhalb von alternativen Hierarchien zu vergeben. Hier verfolgen wir folgendes Konzept:

1.Ein Anwender darf jeden Knoten **auf** und **unterhalb** seiner berechtigten Knoten sehen

2.Dies gilt auch bei alternativen Hierarchien, d.h. wenn ein Knoten für den jew. User sichtbar ist, kann dieser auch in der alternativen Hierarchie die "Kinder" des jew. Knotens sehen.

Wenn z.B. Prof.'in Schulze im Personal-Modul nur die eigenen Kostenstellen sehen darf, im KLR-Modul aber die ganze Fakultät (z.B. wenn sie zeitweise die Rolle einer KLR-Beauftragten in der Fakultät innehält), dann könnte der Administrator bzw. Controller eine spezielle Hierarchie aufbauen, z.B. mit dem Namen "KLR-Baum für Prof.'in Schulze", und diesen Baum bzw. diese Sicht der Frau zuweisen. In dieser Sicht ist der Baum ganz anders aufgebaut:

Fakultät 1 für Geisteswissenschaften

◦**Prof.'in Schulze**

■Lehreinheit Geschichte

•Institut für Frühgeschichte

◦Prof. Meyer

■Lehreinheit Philosophie

•Institut für Humanistische Philosophie

Dadurch, daß der Person Prof.'in Schulze die eigene Kostenstelle, und in dieser Hierarchie darunter alle Kostenstellen der Fakultät liegen, darf Frau Prof.'in Schulze, wenn sie für diese Hierarchie berechtigt ist, auch alle anderen Kostenstellen der Fakultät sehen. Da die Hierarchie nur im KLR-Modul existiert, sind die "normalen" Kostenstellenrechte in den anderen Modulen nicht tangiert, d.h. im Personal- oder Haushaltsmodul dürfte die Person weiterhin nur ihre "eigene" Kostenstelle sehen.

Durch die Funktionalität, alternative Hierarchien anzugeben und Berechtigungen themenbezogen zu steuern, können also beliebige Berechtigungskonzepte realisiert werden, und der Administrator bzw. Controller kann durch den Aufbau des Baums steuern, wo welche Kostenstelle für wen sichtbar sein soll.

## 3.6 (Abfrage-)Masken entwickeln

Die Abfragemasken liefern die Daten aus den Basissystemen an das SuperX-Frontend aus. Einige Abfragen zur Administration sind im Kernmodul enthalten, die Abfragen zu den Basissystemen sind in den jeweiligen Modulen enthalten. Die Abfragen in der Administration erlauben es, neue Masken anzulegen, zu kopieren und zu löschen. Im Folgenden finden Sie allgemeine Hinweise für die Verwaltung der Masken.

Die Masken lassen sich über UNIX, über Access und in Zukunft über ein Java-Frontend administrieren. Unter UNIX geschieht dies über Scripte. Für Windows-Nutzer gibt es ein Access-Frontend, das sich derzeit im Betatest befindet.

### 3.6.1Maskenverwaltung im SuperX-Applet oder XML-Frontend

Die Masken lassen sich im SuperX-Applet verwalten, weitergehende Möglichkeiten bietet aber das **XML-Frontend** (Möglichkeit der Editierung von großen `text`-Feldern bei Postgres als Datenbanksystem). Nach der Anmeldung haben Administratoren das Recht, Masken zu löschen, zu kopieren und erzeugen. Die einzelnen Felder der Masken lassen sich direkt in der Datenbank oder z.B. mit MS Access verändern. Im Applet sind nur grundlegende Verwaltungsoperationen möglich. Sie sind als Ersatz für die UNIX-Scripte gedacht.

Folgende "Abfragen" zur Maskenverwaltung gibt es im Sachgebiet Administration:

Darunter im Ast "Felder" gibt es noch folgende Abfragen:

Darüberhninaus gibt es (nur unter Postgres) die Masken zur Pflege von Masken bzw. Feldern

- Maske erzeugen
- Maske kopieren
- Maske löschen
- Feld kopieren
- Feld löschen
- Maske suchen
- Feld suchen

**Maske erzeugen**. Hier kann eine neue Maske erzeugt werden, und die wichtigsten Zuordnungen der Maske werden angegeben, z.B. Sachgebiet, Themenbaum-Parent etc. Die Felder der Maske selbst in den dazugehörigen Tabellen (z.B. maskeninfo) werden nicht gefüllt oder im Applet administriert. Dazu dienen die Datenbank-Frontends selbst. (s.u.).
Bei der Nummer der Maske (tid) sollten Sie das Nummernschema von SuperX einhalten, um in Zukunft Abfragen-Pooling zu ermöglichen.

**Maske kopieren**. Wie im UNIX - Script wir eine Maske in eine neue Maske kopiert, und alle zugehörigen Tabellen werden aktualisiert. Zusätzlich wird auch der Eintrag im Themenbaum gemacht.

**Maske löschen**. Wie im UNIX-Script werden Masken aus allen dazugehörigen Tabellen entfernt. Zusätzlich wird auch der Eintrag im Themenbaum gelöscht. Zur Sicherheit muss die Nummer der Maske manuell eingegeben werden.

**Maske suchen**. Sie können Masken suchen und im XML-Frontend komfortabel editieren. Schränken Sie Ihre Auswahl auf ein Sachgebiet ein, und drücken Sie "Abschicken". Sie erhalten eine Liste mit "Treffern", und rechts befinden sich jeweils Buttons zum ansehen bzw. editieren einer Maske. Die Maske läuft nur unter Postgres, weil Informix kein direktes Bearbeiten von Blob-Feldern mit sql unterstützt.

**Feld suchen**. Sie können analog zu "Maske suchen" auch Felder suchen und bearbeiten.

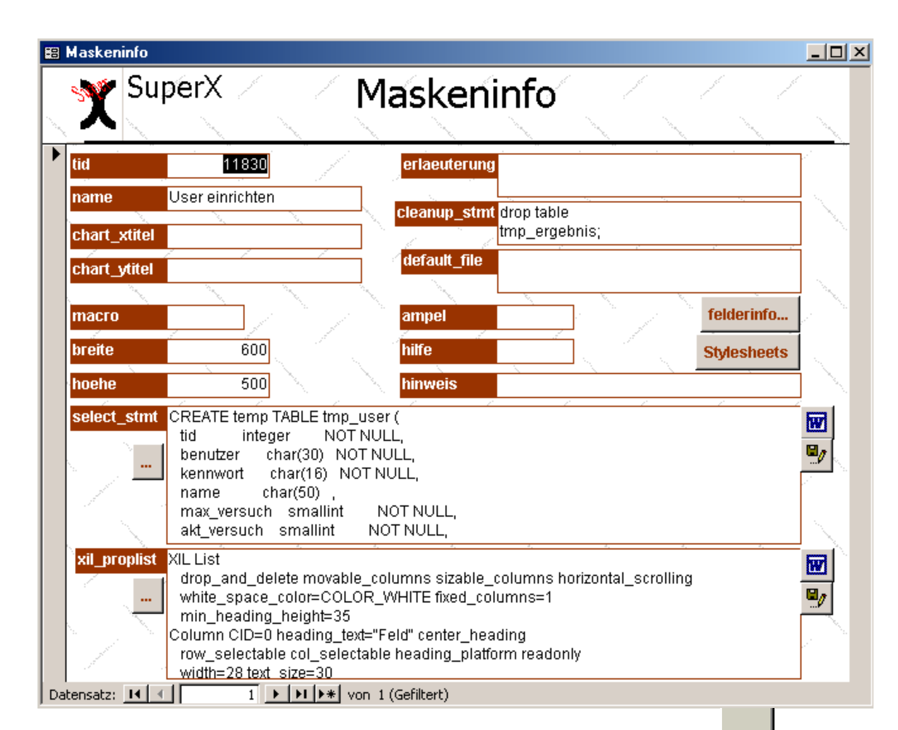
Die Abfragen sind selbsterklärend; das Erzeugen neuer Masken, Löschen vorhandener Masken und Kopieren vorhandener Masken ist nur für Userkennungen möglich, die in der Tabelle `userinfo` im Feld `ad-`

`ministration` den Wert 1 haben. Natürlich sollten die Abfragen sehr vorsichtigt benutzt werden, sie sind die einzigen Abfragen in SuperX, die tatsächlich Änderungen an der Datenbank vornehmen können.

### 3.6.2Maskenverwaltung mit MS Access (obsolet)

Das Access-Frontend ermöglicht die bequeme Änderung von Abfragen (für die Eingabe neuer Masken und Felder empfehlen wir eher die Abfragen im normalen Themenbaum). Es befindet sich im Super-X-Clientpaket in `tools\access\superx_frontend_sam.mdb`. Benennen Sie die Datei um nach `superx_frontend.mdb`. Danach können Sie unter *Masken* die einzelnen Masken von SuperX anwählen und öffnen. Sie erhalten im Formular `maskeninfo` ein Formular, das Eingaben oder Änderungen in der Tabelle `maskeninfo` ermöglicht.

Das Formular ermöglicht es, Masken zu ändern und zu erzeugen. Sie können eine TID vergeben und einen Namen eintragen.




Das `select_stmt` ist ein großes Textfeld und läßt sich besser durch Drücken der ...-Taste in einem separaten Fenster bearbeiten. Leider werden Tabulatoren im normalen Windows-Editor nicht korrekt dargestellt, deshalb befinden sich rechts noch zwei Buttons, mit denen Sie Masken in Word[21] editieren können.

Mit dem Button öffnen Sie das `select_stmt` in Word, und können dort Änderungen vornehmen. Mit dem Button speichern Sie die Änderungen in der Datenbank, und Word wird geschlossen. Bitte be-

[21] Warum ausgerechnet Word? Das Access-Frontend ist in Visual-Basic-for-Applications programmiert, und nach unserer Erfahrung ist dies der am meisten verfügbare Editor mit VBA-Unterstützung, wenn auch Access (als Teil von MS Office) installiert ist. Der Editor WordPad z.B. bietet keine VBA-Schnittstelle. Uns war außerdem eine ausgefeilte Such- und Undo-Funktion wichtig. Theoretisch könnte man in der mitgelieferten Dokumentvorlage `editblob.dot` im gleichen Verzeichnis auch Autotexte und Makros hinterlegen. Daher: Auch wenn es ungewöhnlich ist, Word als IDE zu benutzen: nach unserer Erfahrung ist es recht praktisch. Fehlt nur noch die farbige Syntaxunterstützung...

achten Sie, dass Sie die Dateien in Word nicht speichern müssen. Analog können Sie verfahren, wenn Sie das Feld `xil_proplist` bearbeiten. Um in Access sicherzustellen, dass Feldänderungen wirklich in der Datenbank gespeichert werden, sollten Sie sich einen Button zum Speichern von Datensätzen in die Access-Symbolleiste setzen (Extras -> Anpassen -> Befehle -> Datensatz speichern in eine häufig benutzte Symbolleiste ziehen).

Mit dem Button `Felderinfo` gelangen Sie zu den Feldern dieser Maske. Sie können die Felder dort bearbeiten. Bein Hinzufügen neuer Felder müssen Sie allerdings die jeweiligen `tids` manuell in die Tabelle `masken_felder_bez` eintragen.

Analog funktioniert die Bearbeitung der individuellen Stylesheets für eine Maske.

### 3.6.3Effizientes Debugging

Ein großer Nachteil des 'alten' SuperX war die unvollständige Übermittlung von Fehlermeldungunen bei der Ausführung von Abfragen. Mit dem neuen SuperX gestaltet sich die Entwicklung von Abfragen wesentlich einfacher. Es gibt mehrere Wege, Abfragen zu enwickeln und zu debuggen.

Als Entwickler sollten Sie sich eine einene Servlet-Engine mit Tomcat lokal installieren und das Logging in der Datei
`$SUPERX_DIR/webserver/tomcat/webapps/superx/web-inf/web.xml`
aktivieren (Achtung: dies sollten Sie nur bei lokalen Tomcats tun, nicht im Echtbetrieb, da sonst die Performance leidet). Durch das Logging können Sie genau sehen, an welcher Stelle eine Abfrage abbricht, u.U. übermittelt der JDBC-Treiber auch die Fehlermeldung. Sie können die Protokollierung auch in eine Datei umleiten. So können Sie auch einzelne SQL-Statements aus der LOG-Datei kopieren und in `dbaccess` von Hand ablaufen lassen.

Viele Abfragen in SuperX arbeiten mit temporären Tabellen. Diese sind zwar unter Informix kennungs- und sitzungsabhängig ist, aber es kann beim Abbruch einer Abfrage passieren, daß temporäre Tabellen erhalten bleiben. Beim nächsten Start der Abfrage führt dies also zu einer Fehlermeldung, wenn die temporäre Tabelle neu erzeugt werden soll. So kann es also passieren, dass kein User eine Abfrage mehr ausführen kann, oder gar dass Tomcat abstürzt. In diesem Falle muss man Tomcat einmal beenden und wieder neu starten. Auch deshalb ist es im Echtbetrieb also ratsam, für die Entwicklung von Abfragen einen "eigenen" Tomcat lokal zu installieren.

Ein weiterer einfacherer Weg des SQL-Debugging besteht darin, sich die Fehlermeldungen im Applet anzeigen zu lassen; dazu muss das Logging in der Datei `superx.properties` eingeschaltet werden. Danach können Sie sich die SQL-Statements in der Java-Konsole anschauen. Diese können Sie (unter Windows) in der Systemsteuerung -> Java  Plugin x.x aktivieren. Unter Netscape 6.x mit Linux erreichen Sie die Java-Konsole über das Menü Tools -> Java Console. Sie können auch das SuperX-Applet auspacken (`jar -xvf superx.jar`) und dann SuperX als Anwendung starten mit `java superx >logdatei.txt`. In diesem Falle sehen Sie die Fehlermeldungen direkt auf der Konsole oder in der DOS-Box bzw. in der angegebenen Logdatei.

### 3.6.4Dokumentation von Abfragen: Glossare

Die Statistiken in SuperX ist nicht immer für Außenstehende "selbsterklärend", und insbesondere bei Kennzahlen und kondensierten Werten sollten die Konzepte mit einem Glossar versehen sein.

Die Frontends von SuperX bieten drei Möglichkeiten der Dokumentation:

- Dialogelemente auf den Masken können mit einem "Tool-Tip" versehen werden, d.h. bei Mausbewegung über den Button wird eine Erläuterung angezeigt.
- Ergebnistabellen können mit einem Glossar versehen werden, das die in der Tabelle benutzten Begriffe auf einer zweiten Seite erläutert.
- Umfangreichere Hilfetexte sind über die kontextabhängigen Hilfetexte zu einer Maske und Ergebnistabelle verlinkt. Dies ist in der Entwickleranleitung dokumentiert.

Für die ersten beiden Dokumentationsarten wird in SuperX die Tabelle `sx_captions` gepflegt, die Felderläuterungen und allgemeine Schlüsselörter dokumentieren. Die Dokumentation ist sogar mehrsprachig möglich.

#### 3.6.4.1Allgemeine Schlüsselwörter

Allgemeine Schlüsselwörter sind der Tabelle `sx_captions` definiert, man erkennt sie, daran dass die Spalte id gefüllt ist (table_name, field_name und record_no hingegen leer)

| tid | id | table_name | field_name | record_no | locale | contents_short | contents_long | sachgebiete_id |
|---|---|---|---|---|---|---|---|---|
| **1** | **studiengang** | | | | **de** | **Studiengang** | **Studiengänge definieren sich durch das Fach, die Vertiefungsrichtung, durch Haupt- oder Nebenfach sowie den Abschluss.** | **16** |
| **2** | **studiengang** | | | | **en** | **Subject / Degree** | **A combination of subject and degree as well as the major-minor distinction** | **16** |
| 3 | stud_general | | | | de | Studierende allgemein | | |
| 4 | stud_general | | | | en | students (general) | | |

Im Beispiel wird der Tag **studiengang** definiert.

Dieser Tag wird an beliebiger Stelle (Maskennamen, Überschriften, select_stmt, XIL-Proplist, XSL-Dateien, etc) durch den Eintrag `contents_short` der aktuellen Locale ersetzt.

### 3.6.4.2Der Spezialfall Maskenfelder

Für die Erläuterung von Maskenfeldern können kurze und längere Hilfetexte hinterlegt werden. Die kurzen Texte dienen als Beschriftung des Feldes (überscheiben als den "Feldnamen"), und die langen Texte erscheinen als Tool-Tip bei Mausbewegung auf den Button. Im Ausdruck werden die Maskenfelder wahlweise auf einer separaten Seite dokumentiert.

Damit nicht für jedes einzelne Maskenfeld ein Eintrag gemacht werden muss, kann ein Hilfetext über seinen Namen auch mehreren Maskenfeldern zugeordnet werden; in diesem Fall ist die Spalte `record_no` leer.

Für Felder aus der Tabelle felderinfo schaut SuperX nach, ob in der Tabelle `sx_captions` ein Eintrag für die Tabelle `felderinfo`, `field_name studiengang` und `record_no = 10050` oder `null` vorhanden ist

Im folgenden Beispiel ist ein Maskenbutton "Studiengang" erläutert, der in dieser Weise und bei dem Feld Nummer `10050` dokumentiert sein soll.

| tid | id | table_name | field_name | record_no | locale | contents_short | contents_long | sachgebiete_id |
|---|---|---|---|---|---|---|---|---|
| **9** | | **felderinfo** | **studiengang** | **10050** | **de** | **Grundständiger Studiengang** | **Ein Studiengang im grundständigen Studium** | **16** |
| 10 | | felderinfo | studiengang | 10050 | en | Degree program | | 16 |

Wenn Sie den Erläuterungstext bei allen Feldern mit dem Namen "`studiengang`" erscheinen lassen wollen, dann müssen Sie das Feld `record_no` leer lassen.

## 3.6.5Masken für das XML-Frontend vorbereiten

Das XML-Frontend arbeitet mit den vorhandenen Masken und stellt dort grundlegende Funktionen zur Verfügung. Darüber hinaus bietet das Frontend die Möglichkeit, einzelne Abfragen individuell zu gestalten. Hierzu sind allerdings grundlegende XML-Kenntnisse erforderlich. Außerdem gibt es für den Betrieb gewisse Einschränkungen.

Ein großer Vorteil des XML-Frontends ist, dass Anwender sich ihre Bericht im XML-Format herunterladen können und ohne Datenbankkenntnisse ihre Berichte "maßschneidern" können.

Außerdem arbeitet das XML-Frontend asynchron, d.h. die neuen Servlets können (bislang über die URL) von beliebigen Stellen aus aufgerufen werden. Es ist z.B. damit möglich, auf beliebte Bericht mit gesetzten Parametern einen Bookmark zu legen.

### 3.6.5.1Erzeugen eines Stylesheets

Zunächst muss für das Ergebnis ein neues Stylesheet erzeugt werden. Als Vorlage für Masken können Sie das Muster-Stylesheet

`$SUPERX_DIR/webserver/tomcat/webapps/superx/xml/maske_html_ns.xsl`

(bzw. `maske_html_ie.xsl` mit speziellen Tags für den Internet Explorer von Microsoft) verwenden, für Er-

gebnistabellen können Sie das Muster-Stylesheet
`$SUPERX_DIR/webserver/tomcat/webapps/superx/xml/tabelle_html.xsl`
verwenden. Speichern Sie das Stylesheet unter einem anderen Namen im gleichen Verzeichnis ab, und ändern Sie das Stylesheet. Dann fügen Sie das Stylesheet in die Tabelle `sx_stylesheets` ein.

sx_stylesheets : Tabelle

| tid | filename | caption | description | relation | userag | contenttype |
|---|---|---|---|---|---|---|
| 1 | tabelle_html.xsl | Generisches St | Generisch | table | | text/html; charset=ISO-8859-1 |
| 2 | tabellenfeld_bearbeiten.xsl | Generisches St | Generisch | table | | text/html; charset=ISO-8859-1 |
| 3 | maske_html.xsl | Generisches St | Generisch | mask | | text/html; charset=ISO-8859-1 |
| 4 | maske_html_tabfeld.xsl | Generisches St | Generisch | mask | | text/html; charset=ISO-8859-1 |
| 5 | tabelle_html_11570.xsl | Berichtsblatt K | Kurze Zusamm | table | | text/html; charset=ISO-8859-1 |
| 7 | tabelle_fo_rtf.xsl | RTF | Export in Textve | table | | application/msword |
| 6 | tabelle_fo_pdf.xsl | PDF | Export in PDF ( | table | | application/pdf |
| * | | | | | | |

Datensatz: 1 von 7

Das Beispiel zeigt einige Styleheets, das erste ist bereits Teil des Kernmoduls, das fünfte befindet sich im COB-Modul. Zu den Feldern:

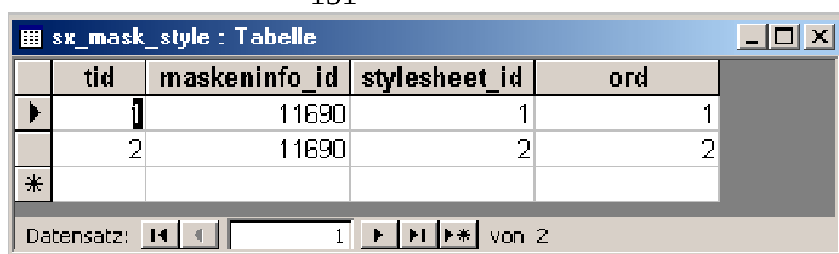
- **filename** kennzeichnet den Dateinamen relativ zum Verzeichnis `$SUPERX_DIR/webserver/tomcat/webapps/superx/xml`.
- **caption** dient als Kurzüberschrift, die im Ergebnisblatt als Button angezeigt wird.
- **description** stellt einen Erläuterungstext für den Button dar.
- **relation** bezieht sich auf die Beziehung des Stylesheets; mögliche Werte sind "mask" für eine Maske und "table" für Tabelle.
- **useragent** beitet die Möglichkeit, ein Stylesheet für spezielle Lesegeräte anzubieten, z.B. WAP-Handys oder Braille-Zeilen.
- **contenttype** entspricht dem useragent und kennzeichnet den `content-type`, der dem Lesegerät im http-header übermittelt werden soll. Möglich sind derzeit die obigen Varianten (svg oder excel sind in Vorbereitung).

### 3.6.5.2Zuordnung einer Maske zu einem Stylesheet

Konkret arbeitet SuperX so: Wenn einer Abfrage ein oder mehrere Stylesheets zugeordnet sind, dann werden die in der Reihenfolge angezeigt, in der sie definiert sind. Wenn kein Stylesheet definiert ist, dann wird das Standard-Stylesheet von SuperX benutzt: `maske_html_ie.xsl` bzw `maske_html_ns.xsl` für Masken sowie `tabelle_html.xsl` für Tabellen.

Die Zuordnung eines Stylesheets geschieht in der Tabelle `sx_mask_style`. Der Tupelidentifier des Stylesheets wird in der Tabelle `sx_mask_style` im Feld `stylesheet_id` eingetragen.

Das Beispiel zeigt, daß die beiden oben beschriebenen Stylesheets der Maske 11690 zugeordnet werden.



| tid | maskeninfo_id | stylesheet_id | ord |
|---|---|---|---|
| 1 | 11690 | 1 | 1 |
| 2 | 11690 | 2 | 2 |

Das Feld ord kennzeichnet die Reihenfolge der anzubietenden Stylesheets. Wir sehen hier, dass zuerst das generische Standard-Stylesheet angezeigt wird, und dann das Stylesheet Nr.2.

Defaultmäßig sind die Stylesheets für html (Druckversion in neuem Fenster), xml und text in jeder Ergebnistabelle enthalten. Die Stylesheets für rtf und pfg müssen in der obigen Tabelle zugeordnet werden - dies ist sinnvoll, da die Standard-Stylesheets zunächst mit der in Frage kommenden Maske erprobt werden muss. Im PDF-Format z.B. muss man die Spaltenbreite nach der Textlänge bestimmen. Und das RTF-Fromat ist aufgrund des "experimentellen" Status von Jfor ebenfalls noch prüfungsbedürftig. In OpenOffice Version 1.1.x ist der erzeugte RTF-Code zum Beispiel unansehnlich, in Microsoft Word dagegen besser.

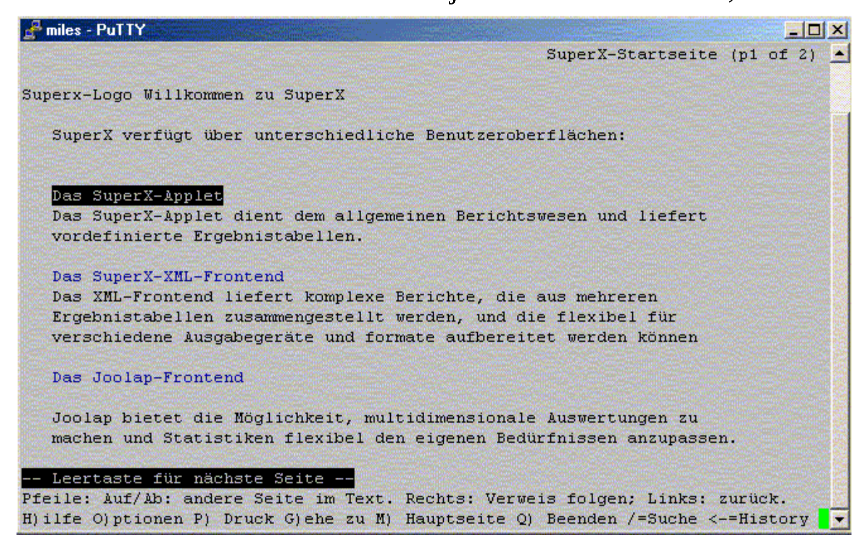
### 3.6.5.3 Anpassung an Lesegeräte

Der Vorteil von XML-Berichten ist, dass sie sich an individuelle Lesegeräte anpassen lassen. So können Sie die Standardoberfläche automatisch für das jeweilige Lesegerät anpassen und dadurch ganz individuelle Designs erzielen, z.B. auch für barrierefreie Angebote.

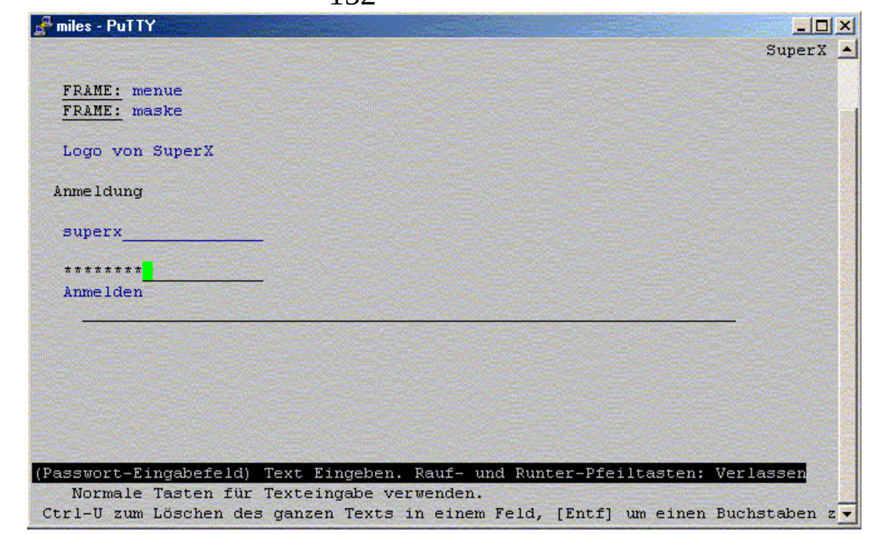
Das folgende Beispiel zeigt dies anhand des textbasierten HTML-Browsers **lynx**, der sich (zumindest am Anfang) gut zum Testen für barrierefreie Angebote eignet.

Klicken Sie jeweils auf die Grafik, um sie zu vergrößern.

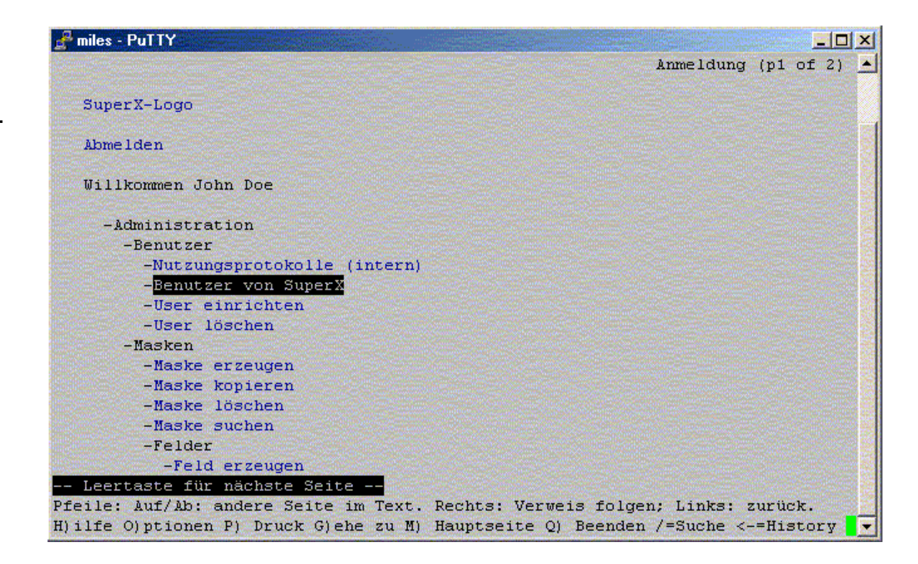
Die rechte Abbildung zeigt die SuperX-Homepage in einer Konsole im Browser lynx.



Wir gehen auf das XML-Frontend, und erhalten die Anmeldemaske. Die Frame-Tags ignorieren wir.



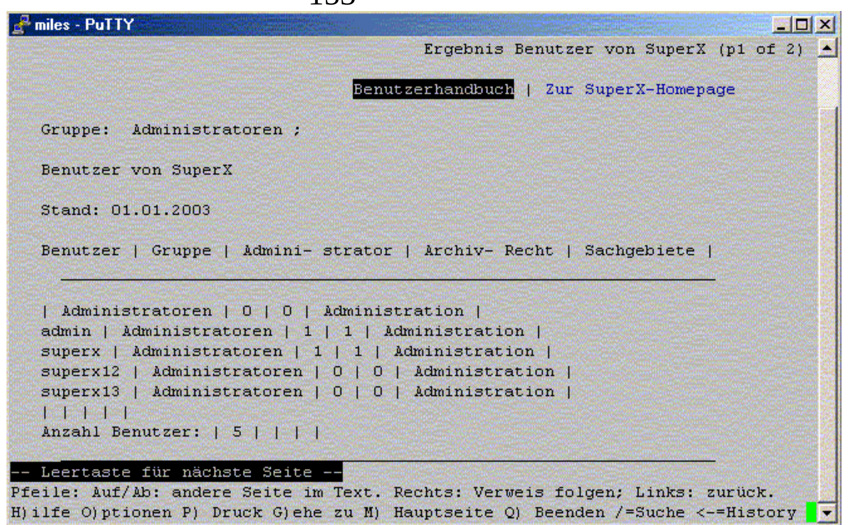

Nach erfolgreicher Anmeldung erscheint das Menü aus dem Themenbaum. Wir wählen hier als Beispiel die Abfrage **Benutzer von SuperX**.




Nun wird die Maske von dieser Abfrage angezeigt. Bei Kombinationsfeldern gehen wir auf das Feld, und drücken die Return-Taste. Es erscheinen die Auswahleinträge. Zum Abschluss gehen wir auf "Abschicken".

miles - PuTTY
Eingabemaske
Zur SuperX-Homepage
Benutzer von SuperX
Benutzer
Gruppe
Sachgebie Administratoren
Thema [ Dezernenten ]
Administr Rektorat/Kanzler
Archiv-Re
Abschicken
(Auswahlliste) Eingabetaste drücken und Pfeiltasten zur Optionswahl verwenden.
H)ilfe O)ptionen P) Druck G)ehe zu M) Hauptseite Q) Beenden /=Suche <-=History

Es erscheint die Ergebnisanzeige. Dies sieht natürlich noch nicht besonders gut aus, weil textbasierte Browser und Tabellen sich nicht gut vertragen. Via Stylesheet lassen sich aber ganz übersichtlich Darstellungen entwerfen.




Das Beispiel zeigt, dass durch XML und XSL keine Grenzen bei der Gestaltung von Benutzeroberflächen für SuperX existieren. Die obigen Stylesheets befinden sich als Muster im Verzeichnis
`$SUPERX_DIR/webserver/tomcat/webapps/superx/xml`.

Wein kleiner Tipp noch für lynx: Wenn Sie das produzierte html überprüfen wollen, dann starten Sie lynx wie folgt:

```
lynx -trace http://localhost:8080/superX/xml/
```

Eine Logdatei `lynx.trace` wird in das aktuelle Verzeichnis geschrieben.

### 3.6.5.4Einschränkungen des XML-Frontends

Das XML-Frontend arbeitet zwar ähnlich wie das Applet, aber es gibt ein paar wichtige Unterschiede. Die Felder einer Maske werden in einem Durchgang aufgebaut, während das Applet die Maskenfelder interaktiv füllt. Dies führt zu folgenden Einschränkungen:

- Es ist im XML-Frontend nicht möglich, im relation-Feld in Felderinfo dynamisch auf den Inhalt eines anderen Feldes (mit <<Feldname>>) zu verweisen.
- Es gibt keine Möglichkeit, den Organigramm-Stand zu verändern
- Der Institut-Button zeigt auch bei der art 4 nur die Einträge an, bei denen "lehre" = 1 gesetzt ist, d.h. die Lehreinheiten und Fakultäten. Alles andere würde zu einer starken Verlangsamung führen (gilt nur bei Organigrammen von über 500 Einträgen).
- Die Mehrfachauswahl ist in html über ein spezielles Listenfeld möglich, dass derzeit aber nur der Internet Explorer unterstützt, nicht Mozilla oder Netscape.
- Das XML-Frontend ist recht langsam und nach unserer Erfahrung auch recht instabil; zukünftige SuperX-Versionen werden sich dieses Problems annehmen.

### 3.6.5.5Erweiterungen des XML-Frontends

Das XML-Frontend bietet gegenüber dem Applet einige Erweiterungen, die insbesondere für aufwändiger gestaltete Webapplikationen nützlich sind:

- Die Ergebnissseiten werden nicht komplett geladen, sondern im Rahmen von frei definierbaren Intervallen, z.B. 30 Datensätze pro Seite. Am Seitenende wird dann eine Navigationsmöglichkeit gelifert (Vorherige Seite / Nächste Seite). Der Intervall wird in `$SUPERX_DIR/webserver/tomcat/webapps/superx/WEB-INF/web.xml` definiert (Parameter `maxOffset`).

- Die Ergebnisseiten können verlinkt werden, über spezielle Navigationsspalten (siehe Entwicklerhandbuch, Kap. "Navigationsspalten im XML-Frontend").
- In Feldern können Links zu anderen Masken definiert werden (Feldart 15).

#### 3.6.5.5.1Export von Abfragen nach PDF und XLS

Im XML-Frontend können Abfragen direkt nach html (Druckversion), XML, PDF oder XLS (->Excel) exportiert werden. Die zugehörigen Stylesheets lauten:

| | |
|---|---|
| **html (Druckversion)** | $SUPERX_DIR/webserver/tomcat/webapps/superx/xml/tabelle_html_p.xls |
| **PDF** | $SUPERX_DIR/webserver/tomcat/webapps/superx/xml/tabelle_fo_pdf.xsl |
| **XLS** | $SUPERX_DIR/webserver/tomcat/webapps/superx/xml/tabelle_xls.xsl |

Der PDF-Konverter arbeiten mit der OpenSource-Bibliothek FOP, der Excel-Konverter mit POI. Die Vorlagen können als Grundlage für eigene Stylesheets verwendet werden. Wir verweisen hier auf der SuperX-Entwicklerhandbuch.

Der PDF-Export funktioniert zwar technisch, aber leider sehen die Ergebnisse oft nicht "schön" aus, da die Berichte in SuperX generell über die Seitenbreite hinaus gehen. Wir empfehlen daher, die Exporte nur bei speziell geeigneten Berichten (mit weniger Spalten) zu verwenden. Außerdem gibt es für Volltexte keine Silbentrennung.

Der XLS-Export wurde mit MS Excel (95-2003) und OpenOffice (1.1.3-2.x) getestet. Da die Produkte automatisch auf Seitenbreite skalieren können, sieht der Export hier deutlich besser aus.

Außerdem können grundlegende Lauyoutelemente wie Kopf- und Fußzeilen und Seitenzahlen individuell angepasst werden, ohne zwingend XSLT-Kenntnisse zu haben.

Schauen Sie dazu im  Abschnitt Individuelle Kopf/Fußzeilen unten.

### 3.6.6Maskensicherung und Rücksicherung im Browser

Mit dem Kernmodul 3.1 bzw. HISinOne 3.0 lassen sich Masken auch browserbasiert entladen und laden.  Dazu wird das Austauschformat XUPDATE sowie das zugehörige Servlet benutzt. Gehen Sie dazu in den Menüpunkt Webanwendung Manager -> Masken-Sicherung.

#### 3.6.6.1Maskensicherung im Browser

Geben Sie in das Feld "oder Spezialparam:" den Wert "maske" ein. Das Feld "Id:" enthält die Masken-ID der Maske, welche gesichert werden soll. Sie erfahren die ID, wenn Sie die Maus über den jew. Menüpunkt zum Öffnen der Maske halten und unten in der Statusleiste des Browsers auf die URL schauen (die Zahl hinter "tid=").

# XUpdater

SuperX 4.1 (build:11.12.2011 21:57)

enter here

```
<xupdate>
</xupdate>
```

oder Spezialparam: maske
id: 16690
Absenden

Wenn Sie auf "Absenden" klicken, erhalten Sie den Quellcode im Feld "enter here":

# XUpdater

SuperX 4.1 (build:11.12.2011 21:57)

enter here

```
<?xml version="1.0" encoding="UTF-8"?>
<xupdate>

<!-- fuer Maske 16690-->
<themenbaum maskentid="16690" parentname="Studierende"/>
<!--Hier Themenbaumparent eintragen Name z.B: Haushalt-->
<sql>delete from maskeninfo where tid = 16690;</sql>
<sql>delete from themenbaum where maskeninfo_id = 16690;</sql>
<sql>delete from felderinfo where tid in (select felderinfo_id from
masken_felder_bez where maskeninfo_id = 16690);</sql>
<sql>delete from masken_felder_bez where maskeninfo_id = 16690;</sql>
<sql>delete from sachgeb_maske_bez where maskeninfo_id = 16690;</sql>
<sql>delete from maske_system_bez where maskeninfo_id = 16690;</sql>
<sql><![CDATA[insert into maskeninfo (tid,name,chart_xtitel,chart_ytitel)
values (16690,'','Studienfach','Anzahl bzw. Anteil');]]></sql>
<sql><![CDATA[update maskeninfo  set name='Studierende nach Abschlüssen
```

oder Spezialparam:
id:
Absenden

Bitte kopieren Sie den Textinhalt in die Zwischenablage und speichern Sie den Text in einer Textdatei mit der Endung ".xml", z.B. `16690.xml`. Diese Datei besitzt das XUPDATE-Format und Sie können sie anderen Hochschulen zur Verfügung stellen oder z.B. in der Berichtsbörse hochladen. Wenn Ihr Server unter UTF-8 arbeitet, sollte der Editor, den Sie benutzen, UTF-8-fähig sein (z.B. Jedit).

### 3.6.6.2Maske im Browser rücksichern

Wenn Sie eine Maske im XUPDATE-Format vorliegen haben, öffnen Sie die Datei mit einem Texteditor, und kopieren Sie den gesamten Inhalt in die Zwischenablage. Wenn Ihr Server unter UTF-8 arbeitet, sollte der Editor, den Sie benutzen, UTF-8-fähig sein (z.B. Jedit).

Gehen Sie dann in den Menüpunkt Webanwendung Manager -> Masken-Sicherung, und fügen Sie den Textinhalt im Feld unter "enter here" ein.  Hier ein Beispiel:

XUpdater

SuperX 4.1 (build:11.12.2011 21:57)

enter here

```
<?xml version="1.0" encoding="UTF-8"?>
<xupdate>

<!-- fuer Maske 16690-->
<themenbaum maskentid="16690" parentname="Studierende"/>
<!--Hier Themenbaumparent eintragen Name z.B: Haushalt-->
<sql>delete from maskeninfo where tid = 16690;</sql>
<sql>delete from themenbaum where maskeninfo_id = 16690;</sql>
<sql>delete from felderinfo where tid in (select felderinfo_id from
masken_felder_bez where maskeninfo_id = 16690);</sql>
<sql>delete from masken_felder_bez where maskeninfo_id = 16690;</sql>
<sql>delete from sachgeb_maske_bez where maskeninfo_id = 16690;</sql>
<sql>delete from maske_system_bez where maskeninfo_id = 16690;</sql>
<sql><![CDATA[insert into maskeninfo (tid,name,chart_xtitel,chart_ytitel)
values (16690,'','Studienfach','Anzahl bzw. Anteil');]]></sql>
<sql><![CDATA[update maskeninfo  set name='Studierende nach Abschlüssen
```

oder Spezialparam:

id:

Absenden

Die Felder darunter können Sie leer lassen, drücken Sie dann direkt auf "Abschicken". Die Maske wird in Ihr System eingespielt.

Danach müssen Sie einmalig im Webanwendung-Manager den Cache leeren und sich dann ab- und neu anmelden. Danach ist die neue Maske unter dem Menüpunkt, der in der Auslieferung vorgegeben wird, sichtbar, z.B. "Studierende".

| Wichtiger Hinweis: die Änderung findet ausschließlich in der Datenbank statt, nicht im Dateisystem Ihres Servers. Wenn von der jew. Maske ein Auslieferungszustand existiert (z.B. die Masken der Komponente "Studierende"), dann wird die Maske beim Upgrade der regulären Komponente ggf. wieder auf den Auslieferungszustand zurückgesetzt. Das XUPDATE-Format bietet sich also nur in den Fällen an, in denen Sie direkt mit den Softwareherstellern in Kontakt stehen, oder die Berichtsbörse für individuelle Berichte nutzen. |
|---|

# 3.7Individuelle Kopf/Fußzeilen

## 3.7.1 Einfache Variante: nur Hochschulename,URL und Logo

Bei der HTML-Darstellung und dem PDF-Export ist auf der ersten Seite im Kopf vorgesehen den Hochschulnamen und die Internetadresse der Hochschule auszugeben. Dies geschieht allerdings nur, wenn diese im System hinterlegt sind. Um die Daten einzugeben oder zu ändern, gehen Sie in der Oberfläche in die Maske "Administration -> Masken -> Beschriftungen suchen". Dort geben Sie bei Stichwort "REPORT" ein.

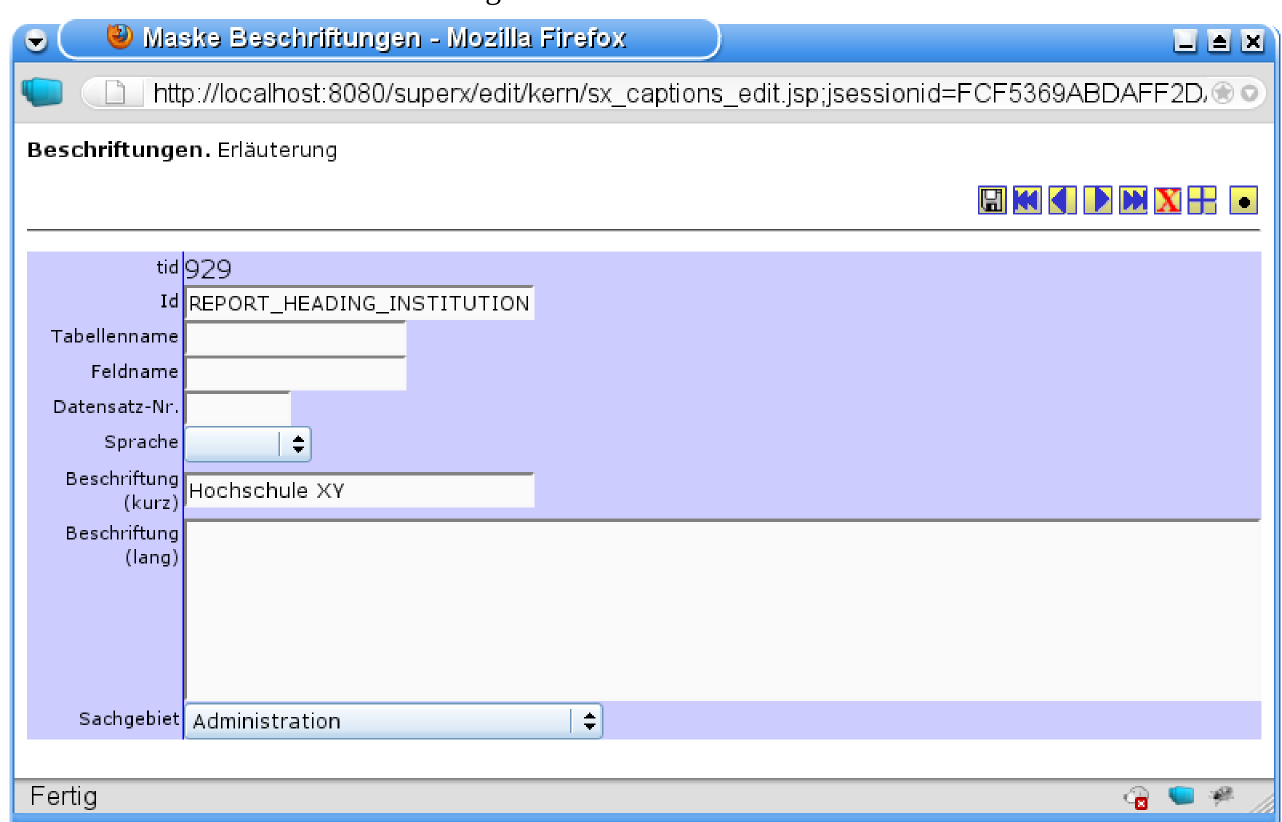
Feld Sprache muss leer sein.

In der Ergebnistabelle sollte nun in der Spalte id jeweils ein Eintrag für "REPORT_HEADING_INSTITUTION" und für "REPORT_HEADING_URL" erscheinen.

Bei "REPORT_HEADING_INSTITUTION" sollte in der Spalte Inhalt (kurz) der Hochschulname stehen und bei "REPORT_HEADING_URL" die Internetadresse der Hochschule.

| id | Tabelle | Feld | Datensatz Nr. | Sprache (kurz) | Inhalt (kurz) | Inhalt (lang) | Bearbeiten |
|---|---|---|---|---|---|---|---|
| REPORT_DOCUMENTATION_URL | | | | | Hochschule | | |
| REPORT_EMAIL | | | | | Hochschule | | |
| REPORT_HEADING_ADRESS | | | | | Hochschule | | |
| REPORT_HEADING_INSTITUTION | | | | | Hochschule XY | | |
| REPORT_HEADING_URL | | | | | www.hochschule-xy.de | | |
| REPORT_LOGO_FILE | | | | | Hochschule | | |

Dies kann über den Button Bearbeiten geändert werden.





Hier kann bei Beschriftung (kurz) der Inhalt geändert werden. Danach einfach auf Speichern (oben rechts) klicken.Der Hochschulname ist schon voreingestellt.

Dann leeren Sie den SuperX Manager Dann leeren Sie den SuperX Manager Cache

**URL der Hochschule**

Soll im HTML/PDF-Kopf auch die URL der Hochschule angezeigt werden, ändern Sie die Variable REPORT_HEADING_URL mit der gleichen Vorgehensweise.

**Eigenes Logo**

Um ein eigenes Logo zu nutzen, spielen Sie die Datei zunächst mit dem Uploadtool hoch (s.u.)

Dann wie gehabt:

Rufen Sie Adminstration/ Masken beschriften auf ,

wählen Sie bei ID REPORT_LOGO_FILE aus und bei Sprache das Feld leeren.

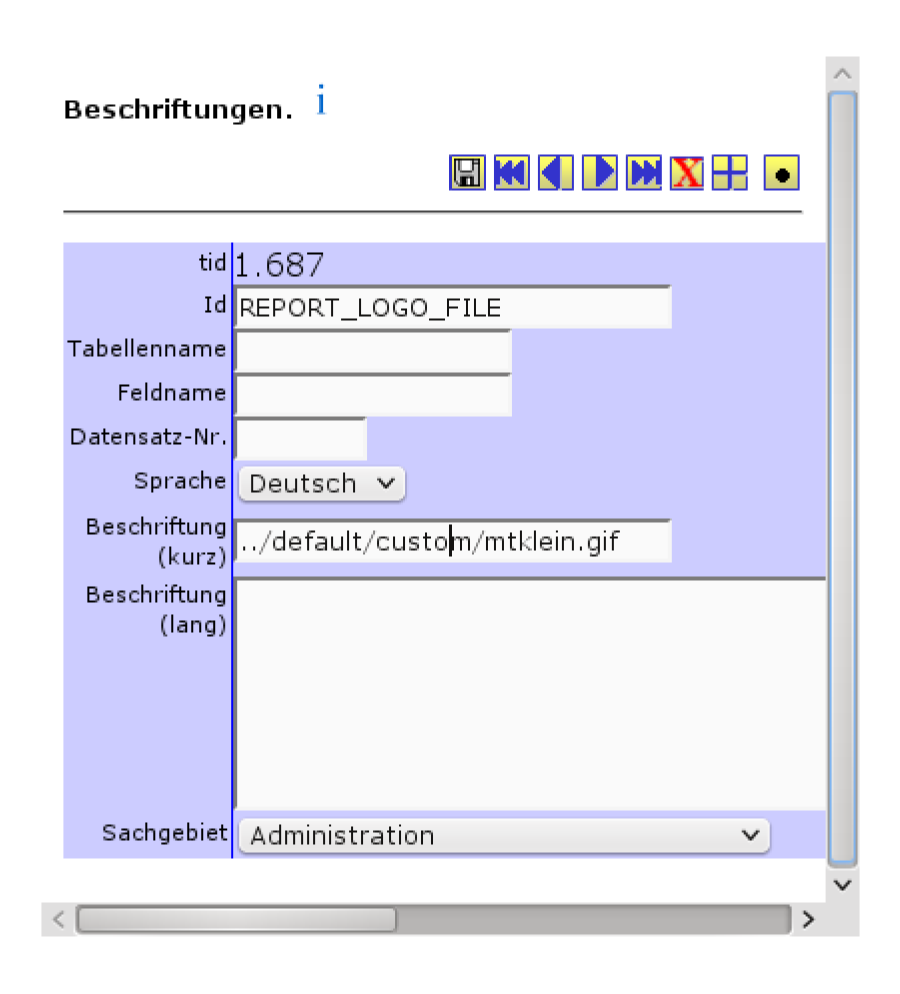
Klicken Sie auf Abschicken und dann bearbeiten.

Machen Sie im Feld "Beschriftung (kurz)" Ihre Änderung und klicken Sie speichern.

Wichtig, die Pfadangabe muss relativ sein, also starten mit

`../MANDANTENID/custom/dateiname`

Statt MANDANTENID muss Ihre konkrete MandantenID angegeben werden, die im Upload-Tool angezeigt wird. Hie rein Beispiel:




Dann leeren Sie den SuperX Manager Cache

Damit wird der Hochschulname ggfs. URL und Logo schon in Standard-HTML und PDF-Berichten angezeigt.

Für Kopfzeilen in Excel gehen Sie nach dem Abschnitt unten vor.

Derzeit noch nicht ausgewertet werden

```
REPORT_HEADING_ADRESS=Adressdaten der Hochschule, oder der Name der
Abteilung, die die Berichte rausgibt

REPORT_EMAIL=Kontaktadresse für Berichte
REPORT_DOCUMENTATION_URL=URL für hochschulinterne Doku-Seiten
```

### 3.7.2 Excel

Um beim Excelexport eine individuelle Kopf/Fußzeile zu nutzen, erzeugen Sie eine Exceldatei mit individueller Kopf/Fußzeile und speichern diese als ExcelVersion bis 2003 ab.

Dateiname vorlage.xls

Laden Sie diese Datei mit dem Uploadtool hoch.

Alternative via XSL:

Schauen Sie in die für das Seitenformat in die Datei `$SUPERX_DIR/webserver/tomcat/webapps/superx/xml/pageComponents.xsl`; dort wird im Abschnitt <xls-page-format> definiert, welche Einträge wie gefüllt werden sollen.

Das rechte Beispiel zeigt die Vorgabewerte des Seitenformats einer Excel-Datei. Die Knoten und Attributnamen sollten selbsterklärend sein.

```
- <xsl:template name="xls_PageFormat">
  - <xls_PageFormat>
      <!--  fontSize="12" klappt leider noch nciht  -->
      <xls_header-left />
      <xls_header-center>SuperX</xls_header-center>
    - <xls_header-right font="Times" fontStyle="normal">
        <xsl:call-template name="today" />
      </xls_header-right>
      <xls_footer-left />
      <xls_footer-center />
      <xls_footer-right font="Times" fontStyle="Italic">Seite $PAGE von $NUMPAGES</xls_footer-right>
      <xls_Autobreaks>true</xls_Autobreaks>
      <!-- wenn nicht gewünscht, dann bitte ganzen Tag entfernen  -->
      <xls_FitHeight>100</xls_FitHeight>
      <!-- soll Tabelle in Höhe auf X Seiten passen? Default ist 100, statt eingebaut 1   -->
      <xls_FitWidth>1</xls_FitWidth>
      <!-- soll Tabelle in Breite auf eine Seite passen?  -->
      <xls_Landscape>true</xls_Landscape>
      <!-- wenn nicht gewünscht, dann bitte ganzen Tag entfernen  -->
      <xls_PaperSize>A4</xls_PaperSize>
      <!-- möglich ist auch "letter"   -->
    </xls_PageFormat>
  </xsl:template>
```

Sie können den gesamten Knoten kopieren in die hochschulspezifische `pageComponents_final.xsl`-Datei, sie überlagert automatisch die "normale" `pageComponents.xsl` und ist sofort in allen Berichten aktiv. So können Sie z.B. im Berichtskopf den Hochschulnamen und die Abteilung eintragen.

### 3.7.3 ganz individuelle HTML-Kopf/Fußzeilen

Einfach nur den Hochschulnamen/URL und Logo anzeigen, geht am einfachsten wie im ersten Abschnitt beschrieben.

Um eine ganz individuelle Kopfzeile für die Berichte im Browser anzulegen, erzeugen Sie testweise lokale auf Ihrem Computer eine Datei htmlheader.htm, die Sie in Ihrem Browser testen können.

(Die Datei braucht nicht auf den Server gespielt zu werden, nur für Sie lokal zum Ausprobieren)

Sie können diese Datei mit HTML gestalten.
Ein einfaches Beispiel
Sie wollen einfach nur den Namen Ihrer Hochschule zentriert über der Tabelle stehen haben,
der Inhalt der Datei kann dann so aussehen.

```
<h2 align="center">Hochschule XY</h2>
```

Wollen Sie zusätzlich ein Logo einbinden, laden Sie das Logo mit dem Upload-Tool hoch und definieren htmlheader.htm z.B. so:

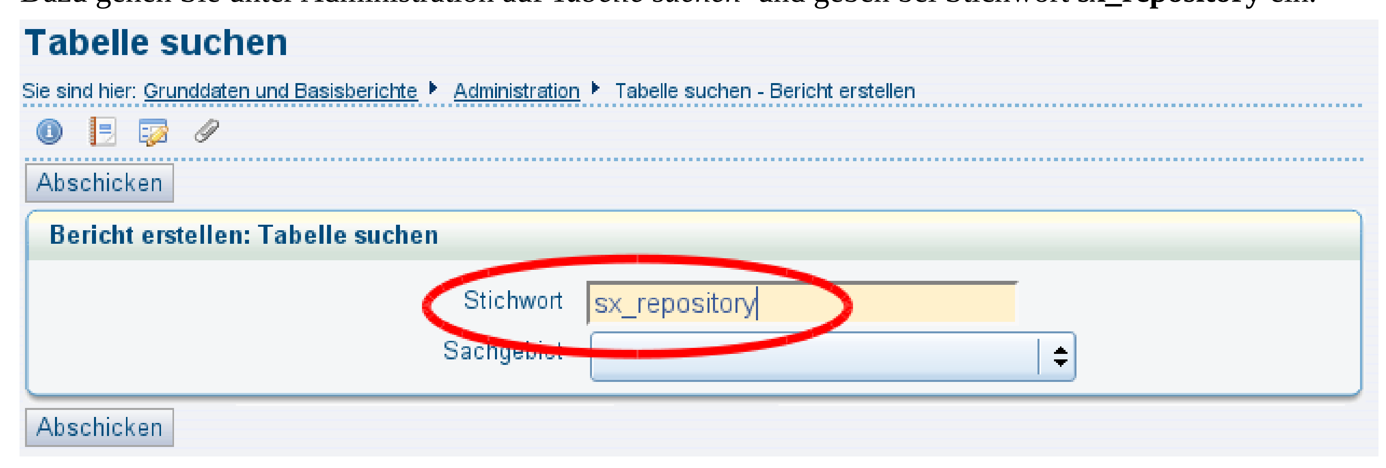
```
<h2 align="center">Hochschule XY<br/>
<img src="/superx/MANDANTENID/custom/logo.gif"></h2>
```

Statt MANDANTENID nehmen Sie Ihre konkrete MandantenId, die Ihnen im Upload-Tool angezeigt wird.

Wenn die Datei gut aussieht, kopieren Sie den Inhalt in die Zwischenablage.
Allerdings ohne die <html> <body> </body> </html> Tags.

Dann wird der Inhalt ins Repository als Variable HTML_HEADER eingespielt.

Dazu gehen Sie unter Administration auf *Tabelle suchen* und geben bei Stichwort **sx_repository** ein.



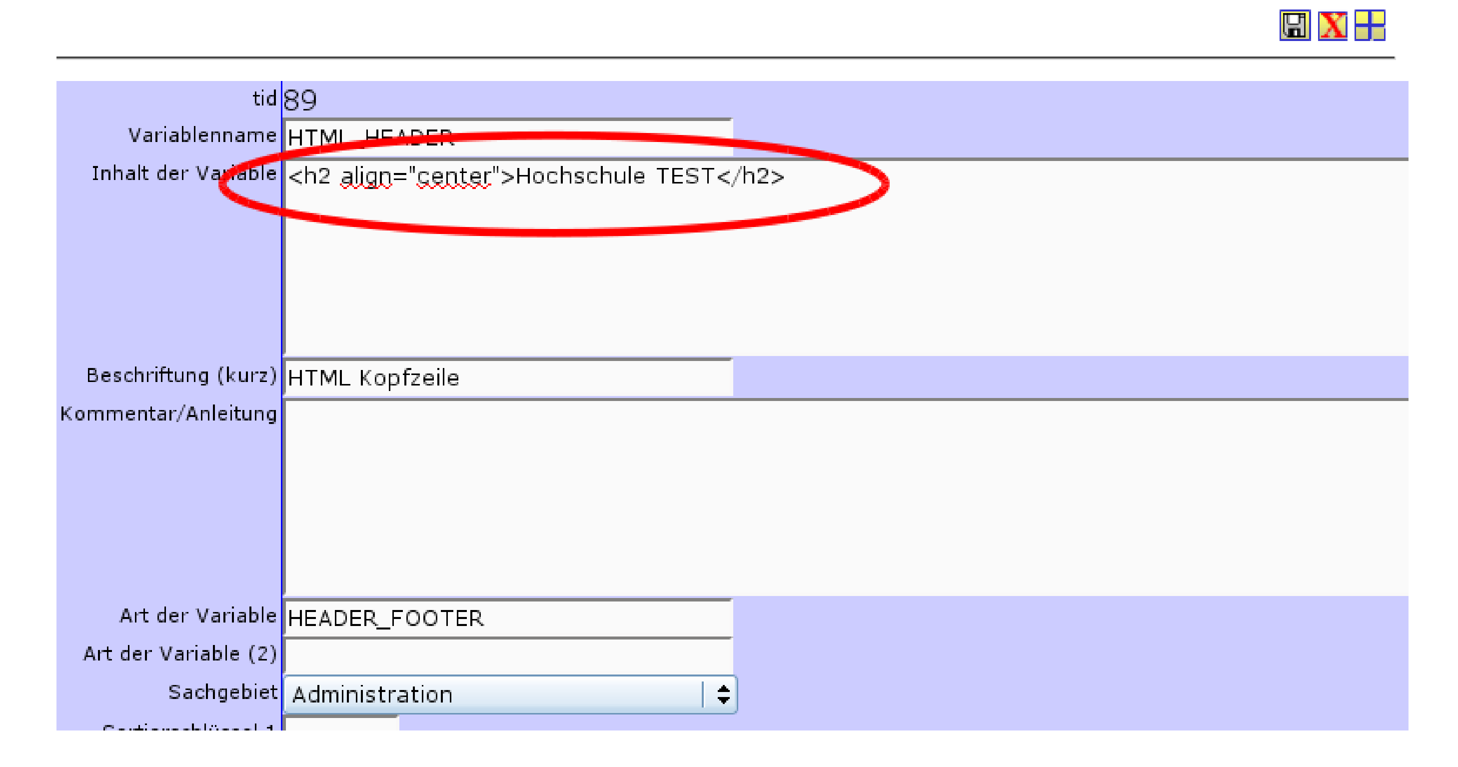

Nach dem Abschicken, klicken Sie bei *sx_repository_list* auf **Bearbeiten.**

In der Zeile mit HTML_HEADER klicken Sie auf Details.

| | | | |
|---|---|---|---|
| 90 | HTML_FOOTER | HTML Fußzeile | HEADER_FOOTER |
| 89 | HTML_HEADER | HTML Kopfzeile | HEADER_FOOTER |

(Falls die Variable HTML_HEADER nicht gefunden wird, leeren Sie einmal den SuperXManager-Cache und versuchen es erneut.)

Tragen Sie Ihre Vorlage bei *Inhalt der Variable* ein, speichern Sie und leeren Sie den SuperX-Manager-Cache.



Für eine individuelle Fußzeile legen Sie nach dem gleichen Prinzip eine Datei htmlfooter.htm an und testen Sie sie lokal.
Sie könnten Sie z.B. füllen mit

```
<p align="center">Datenschutzbestimmungen beachten</p>
```

Dann steht der Hinweis "Datenschutzbestimmungen beachten" unter den Ergebnistabellen im Browser.
Spielen Sie diese als Variable "HTML FOOTER" im Repository ein.

### 3.7.4PDF

Einfach nur den Hochschulnamen/URL und Logo anzeigen, geht am einfachsten wie im ersten Abschnitt beschrieben.

Wenn Sie ganz eigene Kop/Fußzeilen entwerfen wollen:
Für die ganz freie Erstellung individueller Kopf-/Fußzeilen muss man eine XSL-Vorlage erstellen und unter der Variablen CUSTOM_PDF ins repository einspielen.
Dazu gehen Sie unter Administration auf *Tabelle suchen* und geben bei Stichwort **sx_repository** ein.
Nach dem Abschicken, klicken Sie bei *sx_repository_list* auf **Bearbeiten.**
In der Zeile mit **CUSTOM_PDF** klicken Sie auf Details.
(Falls die Variable CUSTOM_PDF nicht gefunden wird, leeren Sie einmal den SuperXManager-Cache und versuchen es erneut.)
Tragen Sie Ihre Vorlage bei *Inhalt der Variable* ein, speichern Sie und leeren Sie den SuperX-Manager-Cache.

Bei PDF kann man separat steuern, wie die Kopf-/Fußzeile der ersten Seite und die der weiteren Seite aussehen soll.
Will man die Höhe einer Kopf/Fußzeile ändern, muss man das an der Stelle machen, wo ein entsprechender Kommentar steht.

z.B.

```
<xsl:template name="first_page_header_height">
<!-- falls Sie die Höhe verändern möchten tragen Sie hier statt 40mm
einen anderen Wert ein -->
<fo:region-before extent="40mm" region-name="first-region-before"/>
```

hier kann man statt 40mm z.B. 60mm eintragen, wenn man ein großes Logo verwendet.
(Der unten im Beispiel angegebene XLM-Header darf nicht fehlen)

Die Gestaltung der Zeilen erfolgt mittels fo bzw. xsl.
Will man z.B. in der Kopfzeile nur den Namen der Hochschule haben, ändert man nach dem entsprechenden Kommentar

```
<!--hier können Sie die Gestaltung der Kopfzeile der ersten Seite an-
passen-->
```

```
<fo:block position="absolute" text-align="center">
<fo:inline font-size="16pt">Hochschule XY</fo:inline>
</fo:block>
</xsl:template>
```

Wichtig ist, dass der <fo:block unten durch ein </fo:block> wieder geschlossen wird und ähnlich der Eintrag <fo:inline> auch mit einem </fo:inline> geschlossen wird.
Möchte man zusätzlich ein Logo einbinden, kann man es mit dem Upload-Tool hochladen und dann z.B. nehmen

```
<fo:block position="absolute" text-align="center">
<fo:inline font-size="16pt">Hochschule XY</fo:inline>
<fo:external-graphic src="../MANDANTENID/custom/logo.gif"></fo:exter-
nal-graphic>
</fo:block>
</xsl:template>
```

Statt MANDANTENID tragen Sie Ihre konkrete MandantenID ein, die vom Upload-Tool angezeigt wird.

Möchten Sie beispielweise in einer Fußzeile
Das Erstellungsdatum, die aktuelle Seitenzahl sowie die Gesamtzahl der Seiten haben, könnte der Eintrag für die Fußzeilen so ausshen

```
<!--hier können Sie die Gestaltung der Fußzeile der ersten Seite an-
passen-->
<fo:block>
<fo:inline align="left" font-size="8pt" space-end="224mm">
Erzeugungsdatum: <xsl:value-of select="/ergebnisse/@datum" />
</fo:inline>
<fo:inline align="right" font-size="8pt">
<fo:page-number />/<fo:page-number-citation ref-id="endofdoc" />
</fo:inline>
</fo:block>
```

Es folgt nun eine komplette Beispieldatei mit den Standardeinstellungen, die Sie als Vorlage nutzen und anpassen können.

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!-- falls UTF-Encoding verwendet wird bei encoding UTF-8 eintagen-->
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:ext1="de.memtext.util.DateUtils"
xmlns:ext2="java.util.Date"
xmlns:string="java.lang.String"
```

```
xmlns:HtmlUtils="de.superx.util.HtmlUtils"
xmlns:fo="http://www.w3.org/1999/XSL/Format">
<!--ERSTE SEITE - KOPFZEILE -->

<xsl:template name="first_page_header_height">
<!-- falls Sie die Höhe verändern möchten tragen Sie hier statt 40mm
einen anderen Wert ein -->
<fo:region-before extent="40mm" region-name="first-region-before"/>
</xsl:template>
<xsl:template name="first_page_header">
<!--hier können Sie die Gestaltung der Kopfzeile der ersten Seite an-
passen-->
<fo:block position="absolute" text-align="end">
<fo:external-graphic>
<xsl:attribute name="src"><xsl:text>servlet/</xsl:text><xsl:call-tem-
plate name="logo_path"></xsl:call-template></xsl:attribute>
</fo:external-graphic>
</fo:block>
</xsl:template>


<!-- ERSTE SEITE - FUSSZEILE -->
<xsl:template name="first_page_footer_height">
<!-- falls Sie die Höhe verändern möchten tragen Sie hier statt 10mm
einen anderen Wert ein -->
<fo:region-after extent="10mm" region-name="first-region-after"/>
</xsl:template>

<xsl:template name="first_page_footer">
<!--hier können Sie die Gestaltung der Fußzeile der ersten Seite an-
passen-->
<fo:block>
<fo:inline align="left" font-size="8pt" space-end="224mm">
Erzeugungsdatum: <xsl:value-of select="/ergebnisse/@datum" />
</fo:inline>
<fo:inline align="right" font-size="8pt">
<fo:page-number />/<fo:page-number-citation ref-id="endofdoc" />
</fo:inline>
</fo:block>
</xsl:template>
```

```
<!-- AB SEITE 2 KOPFZEILE -->
<xsl:template name="rest_header_height">
<!-- falls Sie die Höhe verändern möchten tragen Sie hier statt 10mm einen anderen Wert ein -->
<fo:region-before extent="10mm" region-name="rest-region-before"/>
</xsl:template>
<xsl:template name="rest_page_header">
<!--hier können Sie die Gestaltung der Kopfzeile ab Seite 2 anpassen-->
<fo:block>
<fo:inline align="left" font-size="8pt" > </fo:inline>
</fo:block>
</xsl:template>

<!-- AB SEITE 2 - FUSSZEILE -->
<xsl:template name="rest-region-after-height">
<!-- falls Sie die Höhe verändern möchten tragen Sie hier statt 10mm einen anderen Wert ein -->
<fo:region-after extent="10mm" region-name="rest-region-after"/>
</xsl:template>

<xsl:template name="rest_page_footer">
<!--hier können Sie die Gestaltung der Fußzeile ab Seite 2 anpassen-->
<fo:block>
<fo:inline align="left" font-size="8pt" space-end="224mm">
Erzeugungsdatum: <xsl:value-of select="/ergebnisse/@datum" />
</fo:inline>
<fo:inline align="right" font-size="8pt">
<fo:page-number />/<fo:page-number-citation ref-id="endofdoc" />
</fo:inline>
</fo:block>
</xsl:template>
</xsl:stylesheet>
```

Falls beim Aufruf einer PDF-Datei folgender Fehler kommt:
de.superx.common.DBServletException: Konnte XSL-Datei file:////home/superx/tomcat_sx/webapps/superx/xml/tabelle_fo_pdf.xsl nicht kompilieren
bedeutet dies, dass Ihre CUSTOM_PDF vorlage nicht der XSL-Syntax entspricht. Korrigieren Sie dies ggfs. anhand des Beispiels schrittweise

## 3.8Upload von Dateien per Browser

Für die Gestaltung eigener Kopf/Fußzeilen oder bei der Entwicklung eigener Maskenstylesheets kann es nötig sein, eigene Dateiein auf dem Server zu hinterlegen.

Wenn man Zugriff auf das Dateisystem des Webservers hat, kann man dies natürlich manuell machen, neu ist jetzt aber die Möglichkeit auch Dateien per Browser hochzuladen.

Eigene Dateien werden u.a. aus Sicherheitsgründen in ein eigenes Verzeichnis gelegt
webapps/superx/MANDANTENID/custom
ohne Mandantenbetrieb webapps/superx/default/custom
(falls noch nicht existiert wird das Verzeichnis vom Servlet angelegt)

Um das Upload-Servlet nutzen zu können, muss zunächst die web.xml angepasst werden.

### 3.8.1Anpassung der web.xml

Für das Upload-Servlet sind Ergänzungen in der web.xml nötig (falls noch nicht vorhanden).

Unter servlets

```
 <servlet>
         <servlet-name>SuperXUpload</servlet-name>
         <servlet-class>de.superx.servlet.SuperXUpload</servlet-class>
            <init-param>
            <!--MandantenID-->
            <param-name>default</param-name>
            <!--Filter   *   vorlage.xls,*.xsl-->
            <param-value>vorlage.xls,*.gif</param-value>
            </init-param>
            <init-param>
</servlet>
```

Für jeden Mandanten,der das Upload-Servlet nutzen soll, muss es einen Parameter geben.

Wenn es für einen Mandanten keinen param-Eintrag gibt, kann er das Upload-Servlet nicht benutzen.

Ohne Mandantenbetrieb ist es der param-name einfach nur default wie oben, bei zwei Mandanten FH_TEST1 und FH_TEST2

```
<init-param>
   <param-name>FH_TEST1</param-name>
    <param-value>vorlage.xls,*.gif,*.png,*.jpg,*.htm</param-value>
</init-param>

<init-param>
   <param-name>FH_TEST2</param-name>
    <param-value>vorlage.xls,*.gif,*.png,*.jpg,*.htm</param-value>
</init-param>
```

Als Parameter-Value wird eingetragen, welche Dateien/Dateiarten, die Hochschulen hochladen können sollen.

Weiterhin in der web.xml

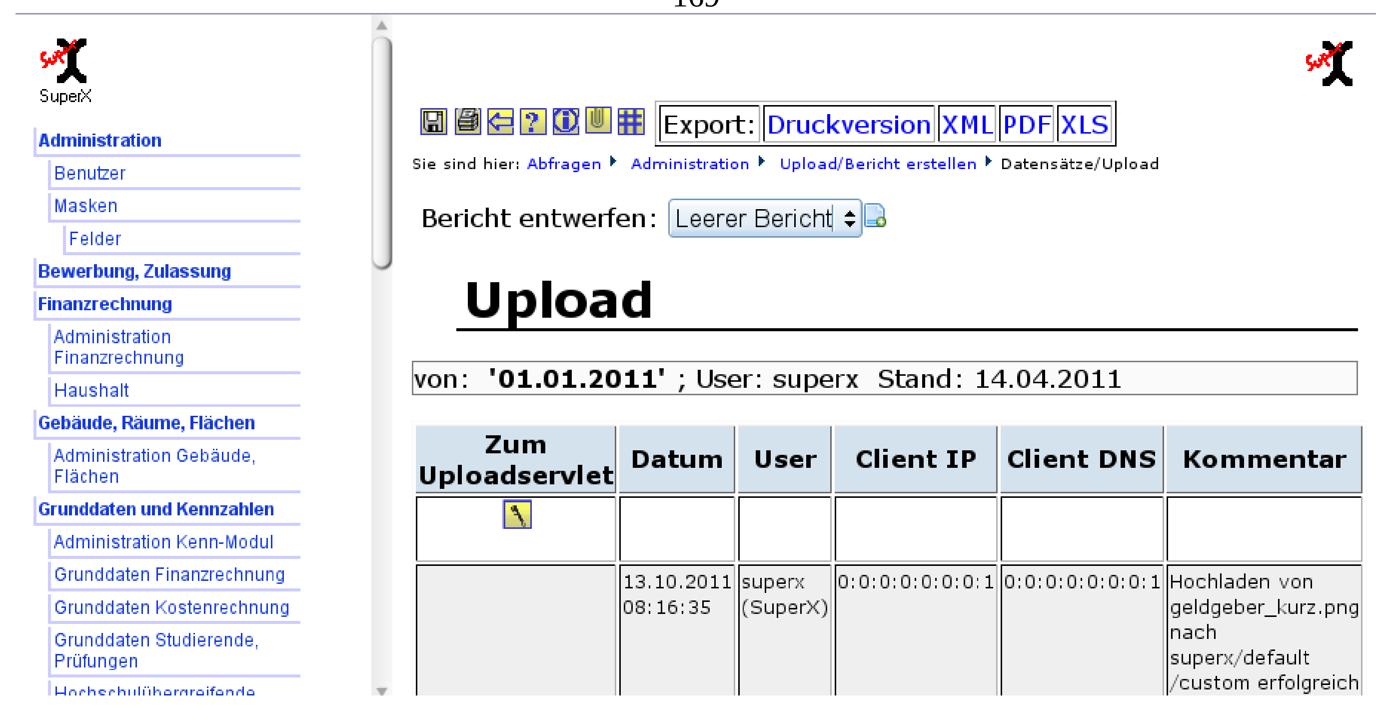
Unter servlet-mapping

```
<servlet-mapping>
        <servlet-name>SuperXUpload</servlet-name>
        <url-pattern>/servlet/SuperXUpload</url-pattern>
   </servlet-mapping>
```

### 3.8.2Nutzen des Upload-Servlets

Rufen Sie unter Administration / **Upload** auf.

Wenn Sie Abschicken anklicken, sehen Sie ein Protokoll über Uploads und ganz oben ist ein Link zum UploadServlet.




Das Upload-Servlet gibt viele Infos aus, wie Ihre MandantenID oder erlaubte Dateien.
Sie können maximal vier Dateien gleichzeitig hochladen.

## SuperX Upload

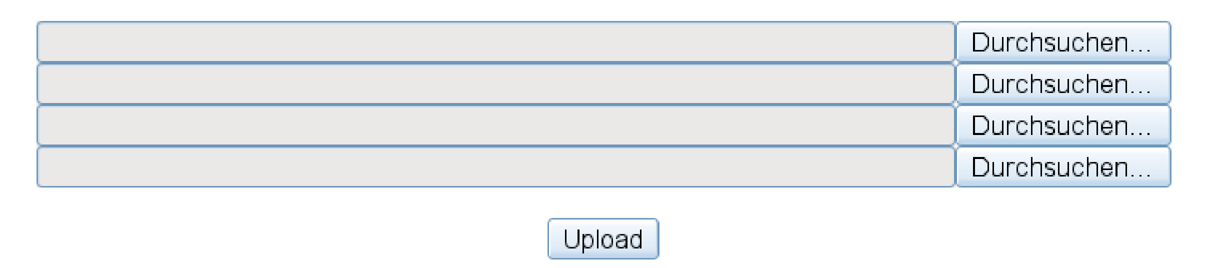
**Ihre MandantenID:** default

Dateien werden auf dem Webserver gespielt ins Verzeichnis **superx/default/custom**
Zur Verlinkung von Grafiken kann z.B. ../custom/logo.gif genutzt werden

**Erlaubte Dateien:** vorlage.xls,*.gif,*.png,*.htm

**Jeder Upload wird protokolliert!**

Sie können ein bis vier Dateien gleichzeitig hochladen.
Ggfs. bereits vorhandene Dateien werden überschrieben.




**Jeder Upload wird einschließlich Dateiname,Username,Zeitpunkt und IP-Nummer protokolliert!**
Nach dem Upload erhalten Sie eine Bestätigung.

**SuperX Upload**

Hochladen von geldgeber_kurz.png nach superx/default/custom erfolgreich

zur Upload-Seite

### 3.8.3 Eigene XSL-Stylesheets mittels Upload-Funktion

Wenn eine Hochschule eigene XSL-Stylesheets mittels upload-Funktion nutzen möchte, sind zwei Dinge zu beachten:

1. Freigabe für *.xsl-Dateien muss in der web.xml eingetragen sein (s.o.)
2. Die Stylesheets werden in das Verzeichnis tomcat/webapps/superx/MANDANTENID/custom geladen, daher müssen in dem Stylesheets Links relativ sein.
   z.B. statt standardmäßig <xsl:import href="xsl_functions.xsl" />
   <xsl:import href="../../xml/xsl_functions.xsl" />
3. filename-Eintrag in SuperX-Tabelle sx_stylesheets muss auch relativ sein,
   z.B. ../custom/MANDANTENID/maske_html_M1.xsl

## 3.9 Embedding SuperX: Eigene Oberflächen für SuperX gestalten

Es ist in SuperX mit dem Kernmodul 3.5 möglich, einzelne SuperX Masken und Ergebnistabellen in eigene Web-Präsenzeen einzubetten. Es werden dabei direkte Hyperlinks auf das SuperX-Servlet genutzt, d.h. unter Umgehung der normalen Menüstruktur in SuperX. Da bei jedem Zugriff die Authentifizierung und die jew. Rechte überprüft werden, ist dies auch sicherheitstechnisch kein Problem.

### 3.9.1 Allgemeines Vorgehen

Wir erzeugen einen HTML-Hyperlink nach dem Muster

http://<<Pfad zum SuperX-Servlet>>?Feld1=Wert1&Feld2=Wert2...

Beim Pfad zum SuperX-Servlet gibt es drei Möglichkeiten:

| | |
|---|---|
| **Pfad zum Menü (Themenbaum)** | http://<<Servername>>:<<Port>>/superx/servlet/SuperXmlAnmeldung |
| **Pfad zur Maske** | http://<<Servername>>:<<Port>>/superx/servlet/SuperXmlMaske |
| **Pfad zur Tabelle** | http://<<Servername>>:<<Port>>/superx/servlet/SuperXmlTabelle |

Der Pfad zum Menü liefert ein HTML-Menü zurück, der Pfad zur Maske eine Maske. Der Pfad zur Tabelle liefert direkt die gewünschte Tabelle. Als erster Parameter sollte bei Masken und Tabellen der Parameter "tid=...." übergeben werden, dies ist die Maskennummer.

Diese Hyperlinks können wir in eine vorhandene Webpräsenz einbauen, Anwender, die (noch) nicht authentifiziert sind, müssen sich beim ersten Aufruf der URL anmelden, und werden dann zur gewünschten Seite weitergeleitet. Wenn mehrere Seiten aufgerufen werden sollen, müssen die Anwender allerdings für den SuperX-Server Cookies erlauben.

Das allgemeine Vorgehen ist sehr einfach, das Problem liegt nur im Detail: wir müssen für die Konstruktion des Hyperlinks die Schlüsel der zu übergebenden Felder kennen, und wir müssen alle Sonder-

und Leerzeichen in Feldnamen oder Werten entfernen / abfangen. Wir können aber zur Erleichterung der Arbeit die Lesezeichen-Funktion oder Schlüsselanzeige von SuperX nutzen.

### 3.9.2Beispiel für eine eingebettete Seite

Nehmen wir an wir wollen direkt auf folgende Seite verlinken:

## Alter der Studierenden

Köpfe oder Fälle ?: **Köpfe** ; Stichtag: **Aktuelle Zahlen** ; Seit Semester: **SS 2006** ; bis Semester: **SS 2006** ; Status: **Alle ohne Beurl.** ; Hörerstatus: **Haupthörer** ; Aggregierung Fach: **Fächer + Studiengänge** ; User: superx Stand: 16.08.2008

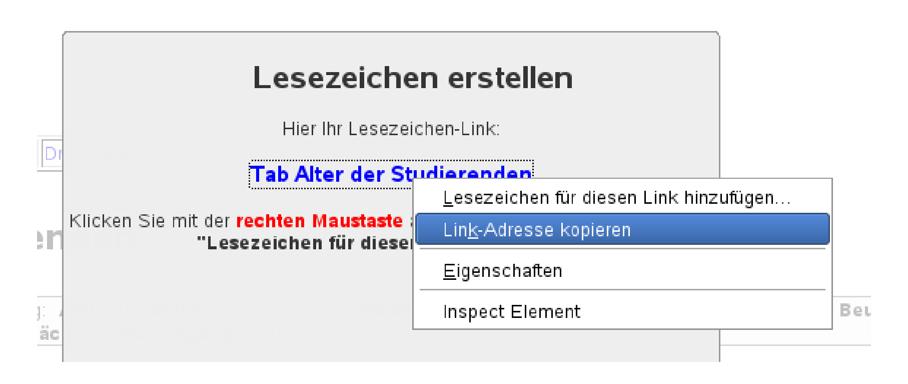
Datensatz 1 - 30 von insgesamt 209 Sätzen

| Ebene | Studiengang | Gesamtzahl | Durchschnitt | <20 | 20-24 | 25-29 | 30-34 | 35-39 | 40-44 | 45-49 | 50-54 | 55-59 | >=60 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Summe Fach (intern)** | **Fach (intern)** | **4.018,00** | **25,64** | **48,00** | **2.297,00** | **1.040,00** | **277,00** | **162,00** | **114,00** | **45,00** | **28,00** | **6,00** | **1,00** |
| Fach (intern) | Biologie | 151,00 | 24,23 | 1,00 | 100,00 | 38,00 | 8,00 | 3,00 | 1,00 | 0,00 | 0,00 | 0,00 | 0,00 |
| Studiengang | Biologie Bachelor VM Prüf.-Ordn. 20052 | 32,00 | 23,00 | 1,00 | 25,00 | 4,00 | 0,00 | 1,00 | 1,00 | 0,00 | 0,00 | 0,00 | 0,00 |

Es wird das Servlet SuperXmlTabelle mit den Parametern "Köpfe oder Fälle = Köpfe" etc. aufgerufen.

Die zugehörige URL können wir durch den Lesezeichen-Button erfahren:

Mit Klick auf den blauen Link mit der Rechten Maustaste können Sie den Link in die Zwischenablage kopieren:



Der Text in der Zwischanablage ist ein Javascript-Aufruf, der den Komfort im normalen Lesezeichen erhöht. Wir benötigen aber nur den fett hervorgehobenen Text, nämlich die URL innerhalb der "":

```
javascript:url="http://mercury:8080/superx/servlet/S
uperXmlTabelle?tid=16340&K%C3%B6pfe%20oder%20F
%C3%A4lle%20%3F=studiengang_nr%20%3D%201%20and
%20fach_nr%20%3D%201&Stichtag=0&Seit
%20Semester=20061&bis%20Semester=20061&F%C3%A4cher-
Sicht=k_stg&Status=1%2C2%2C3%2C5%2C6&H
%C3%B6rerstatus=hrst%3D'H'&Aggregierung%20Fach=10";
if (window.oeffne) oeffne(url); else
self.location.href=url;

-->
```

Diesen Text können wir in einer einfachen HTML-Seite einfügen. Wir müssen nur noch eine Kleinigkeit anpassen: für den Javascript wurden etwaige Umlaute "unescaped", d.h. umcodiert. Wir müssen also in dem Link die fett hervorgehobene Funktion aufrufen.

```
<html>
<body>
<p><a
href="javascript:document.location=unescape('http://mercu-
ry:8080/superx/servlet/SuperXmlTabelle?tid=16340&K%C3%B6pfe
%20oder%20F%C3%A4lle%20%3F=studiengang_nr%20%3D%201%20and
%20fach_nr%20%3D%201&Stichtag=0&Seit%20Semester=20061&bis%20-
Semester=20061&F%C3%A4cher-
Sicht=k_stg&Status=1%2C2%2C3%2C5%2C6&H%C3%B6rerstatus=hrst
%3D'H'&Aggregierung%20Fach=10');" >TEST</a>
</p>
</body>
</html>
```

Analog können Sie auch auf Masken verlinken, die jew. Felder sind dann entsprechend vorbelegt, ermöglichen dem User aber dann, diese oder andere Parameter zu ändern:

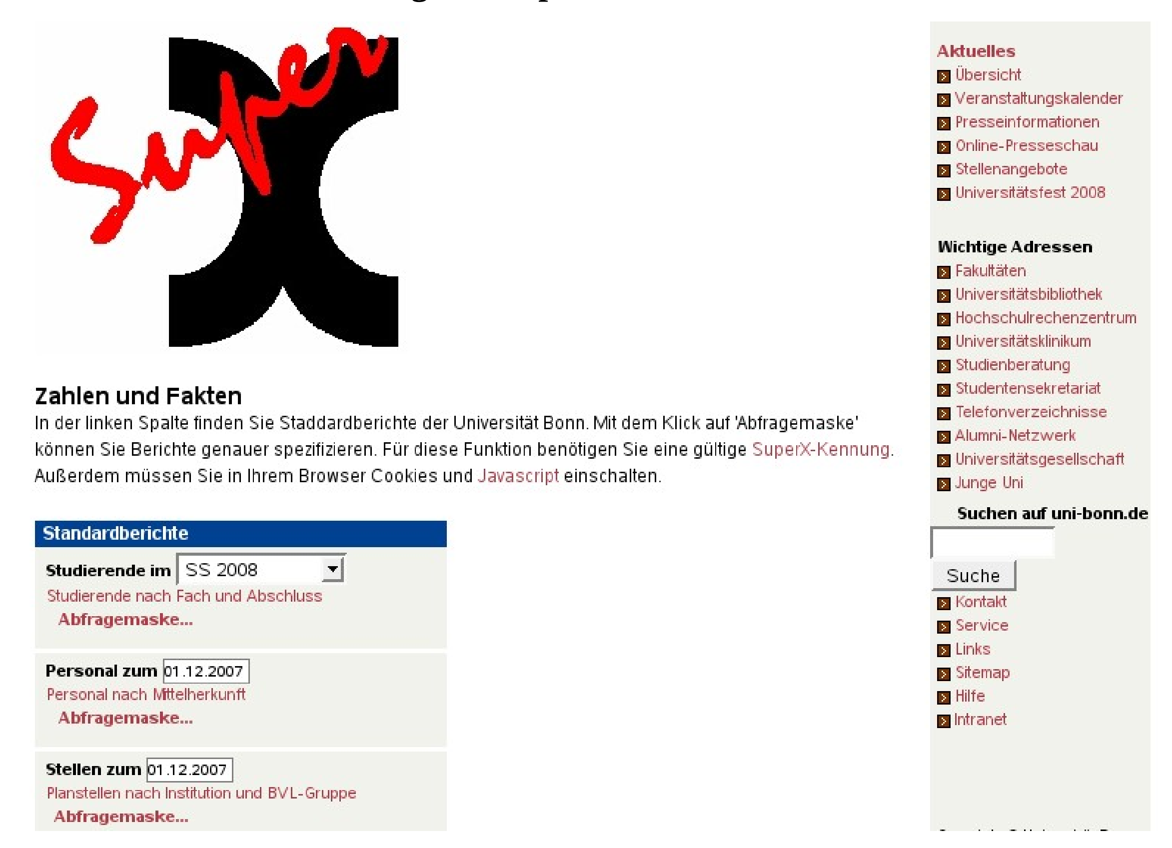
```
<p>Dies ist ein <a href= "javascript:document.location= unescape ('http://mercury:8080/superx/serv-
let/SuperXmlMaske?tid=16340&K%C3%B6pfe%20oder%20F%C3%A4lle%20%3F=studiengang_nr%20%3D%201%20and
%20fach_nr%20%3D%201&Stichtag=0&Seit%20Semester=20061&bis%20Semester=20061&F%C3%A4cher-
Sicht=k_stg&Status=1%2C2%2C3%2C5%2C6&H%C3%B6rerstatus=1%3D1&Aggregierung%20Fach=10');">Test einer
Maske</a></p>
```

Hier ist der Link auf das Servlet `SuperXmlMaske` eingetragen.

### 3.9.3 Komplexeres Beispiel für die Einbettung von SuperX

Oben wurde gezeigt, wie man mit Hilfe einer URL eine Ergebnisseite direkt abrufen kann. Nun ist es hier und da sicher sinnvoll, eigene Parameter in einer benutzerdefinierten Maske anzugeben. Das folgende Beispiel zeigt eine Aufrufseite der Universität Bonn. Zunächst wird die Oberfläche gezeigt, und dann die zugehörige Technik dafür.

### 3.9.3.1Oberfläche der Einbettung von SuperX in vorhandene Websites




Die Seite bietet eine Studierenden- eine Personal- und eine Stellenstatistik, wobei einzelne Parameter (Semester, Datum) vom Anwender variiert werden können. Wenn ein Anwender z.B: bei der Voreinstellung "SS 2008" auf den Link "Studierende nach Fach und Abschluss" klicken, gelangen Sie (beim ersten Mal) zu einem Login-Dialog:

## Sie müssen sich zunächst anmelden

Kennung

superx

Passwort

********

Abschicken

Nach dem Login erscheint direkt die Tabelle für das SS 2008:

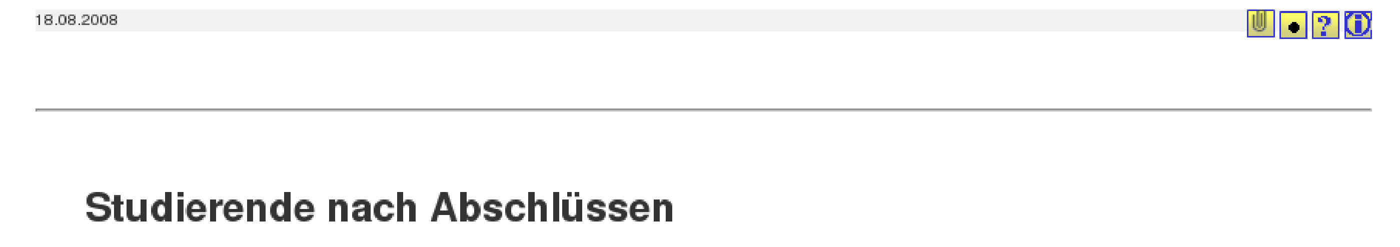
## Studierende nach Abschlüssen

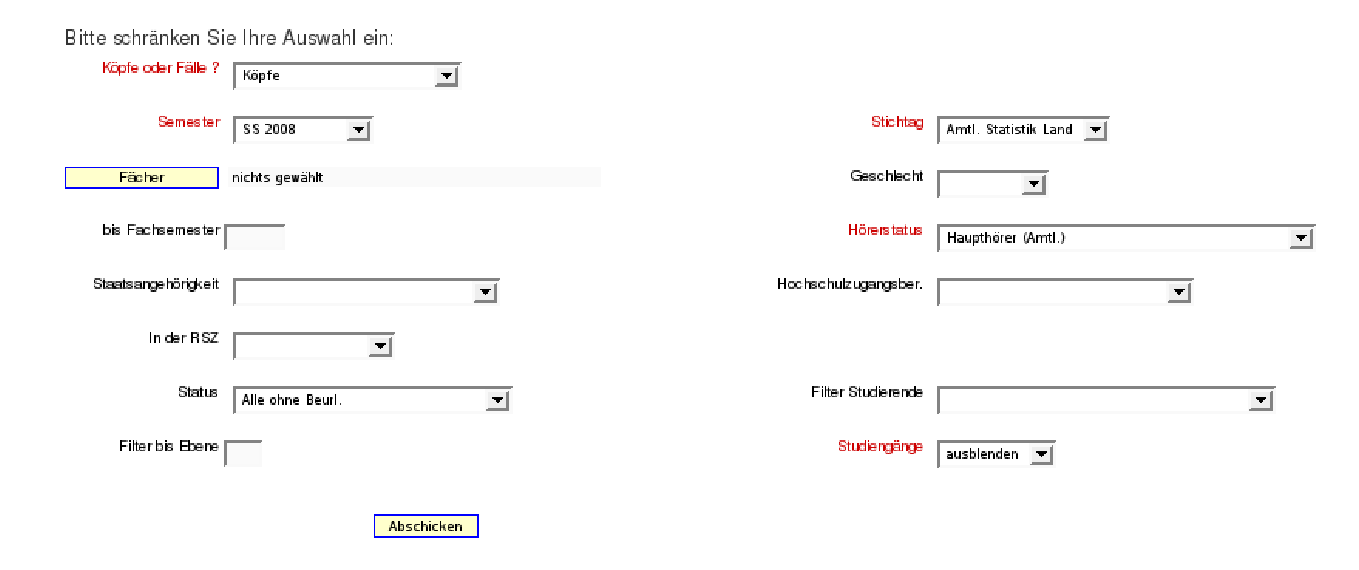
Köpfe oder Fälle ?: **Köpfe** ; Semester: **SS 2008** ; Stichtag: **Amtl. Statistik Land** ; Hörerstatus: **Haupthörer (Amtl.)** ; Status: **Alle ohne Beurl.** ; Studiengänge: **ausblenden** ; User: superx Stand: 07.08.2006

Erster Studiengang, erstes Fach

| Ebene | Art d. Ebene | Studiengang | Gesamt | Diplom HF | Diplom NF | Staatsex. HF | Staatsex. NF | Magister HF | Magister NF | Master HF | Master NF | Bachelor HF | Bachelor NF | Promoti HF |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

Man könnte nun direkt zurückgehen und ein anderes Semester wählen. Man könnte aber auch den Button "Zurück zur Maske" anklicken, dann gelangt man zur Abfragemaske:




### 3.9.3.2Technik der Einbettung von SuperX in vorhandene Websites

Die obige Oberfläche wurde durch eine HTML-Seite erzeugt, in die folgende Komponenten eingefügt wurden:

Eine Javascript-Methode, die den Aufruf einer URL realisiert mit Parameter realisiert:

```
<script language="Javascript" type="text/javascript">
function openWindow(fld_name,fld_val,url)
{
if(fld_val=='')
 alert("Bitte fuellen Sie das Feld "+fld_name+" aus");
 else
 {
zs=url+"&"+fld_name+"="+fld_val;
 window.open(zs,"_blank");
 }
}
</script>
```

Der folgende Quellcode zeigt, wie die Tabelle und darunter die Maske aufgerufen wird:

```
<form name="Studierende"> <b>Studierende im <SELECT class="maskinputPflicht" id="Sem" NAME="Sem"
tabindex="1" ><option class="maskinput" value="20082">WS 2008/2009</option><option class="maskinput"
value="20081" selected="true">SS 2008</option></SELECT></b> <br>
<a
href="javascript:openWindow('Semester',document.forms['Studierende'].elements[0].options[document.for
ms['Studierende'].elements[0].selectedIndex].value,'../../servlet/SuperXmlTabelle?
tid=16280&'+escape('Köpfe oder Fälle ?')+'=studiengang_nr%20%3D%201%20and%20fach_nr%20%3D
%201&Stichtag=1&'+escape('Hörerstatus')+'='+escape('hrst in (select apnr from sos_k_hrst where
astat=\'1\')')+'&Status=1%2C2%2C3%2C5%2C6&'+escape('Studiengänge')+'=2');">
 Studierende nach Fach und Abschluss</a><br>
   <b><a
href="javascript:openWindow('Semester',document.forms['Studierende'].elements[0].options[document.for
ms['Studierende'].elements[0].selectedIndex].value,'../../servlet/SuperXmlMaske?
tid=16280&'+escape('Köpfe oder Fälle ?')+'=studiengang_nr%20%3D%201%20and%20fach_nr%20%3D
%201&Stichtag=1&'+escape('Hörerstatus')+'='+escape('hrst in (select apnr from sos_k_hrst where
astat=\'1\')')+'&Status=1%2C2%2C3%2C5%2C6&'+escape('Studiengänge')+'=2');">
  Abfragemaske...</a>
</b>
</form>
```

Wie Sie sehen werden die Felder mit Sonderzeichen, Hochkommata durch die Javascript-Funktion "escape(...)" umgewandelt, damit der Aufruf korrekt übergeben wird. Das Hochkomma muss darüber hinaus auch durch ein "\" maskiert werden.

Das obige `<form>`-Element wurde mit relativen URLs erzeugt, d.h. Sie können den Code z.B. in eine Seite `http://<<Ihr-Server>>:<<Port>>/superx/xml/public/index.htm` übernehmen, und der Code ist direkt lauffähig.

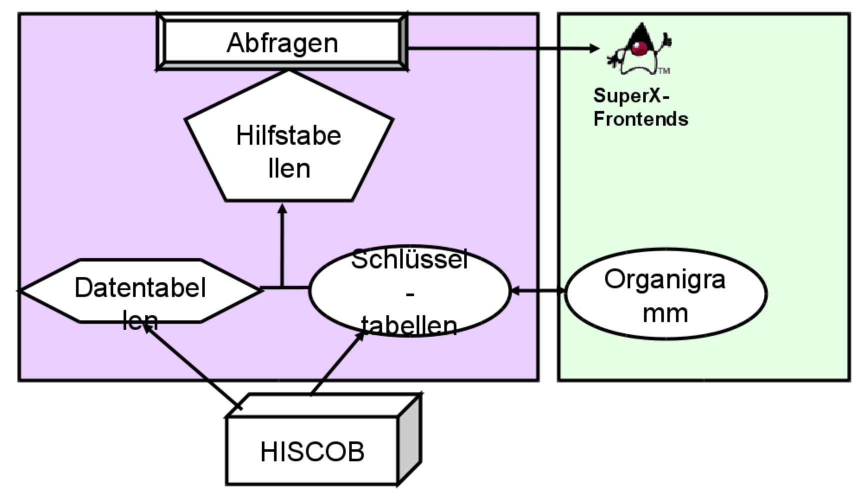
## 3.10 Installation von Modulen

Das Kernmodul enthält außer Administrationsabfragen und Tabellen keinerlei Inhalte. Die Inhalte werden in Form von Modulen hinzugefügt. Dazu gibt es vorgefertigte Installationsscripte.

### 3.10.1 Architektur von SuperX-Modulen

Die folgende Abbildung zeigt die Architektur von Modulen ab Beispiel vom HISCOB-Modul:

Ein Modul besteht auf Datenbankseite aus Abfragen, Hilfstabellen, Datentabellen und Schlüsseltabellen (sowie Prozeduren).
Auf Webserver-Seite können auch XSL-Stylesheets vorhanden sein.




Die Abbildung zeigt, dass ein Modul eigene Komponenten nutzt, aber auch auf Teile des Kernmoduls zugreift, z.B. das Orgranigramm - dies macht SuperX zu einem integrierten System. Neben dem Organigramm sind alle anderen Komponenten des Kernmoduls natürlich betroffen, z.B. Themenbaum, Userrechte.

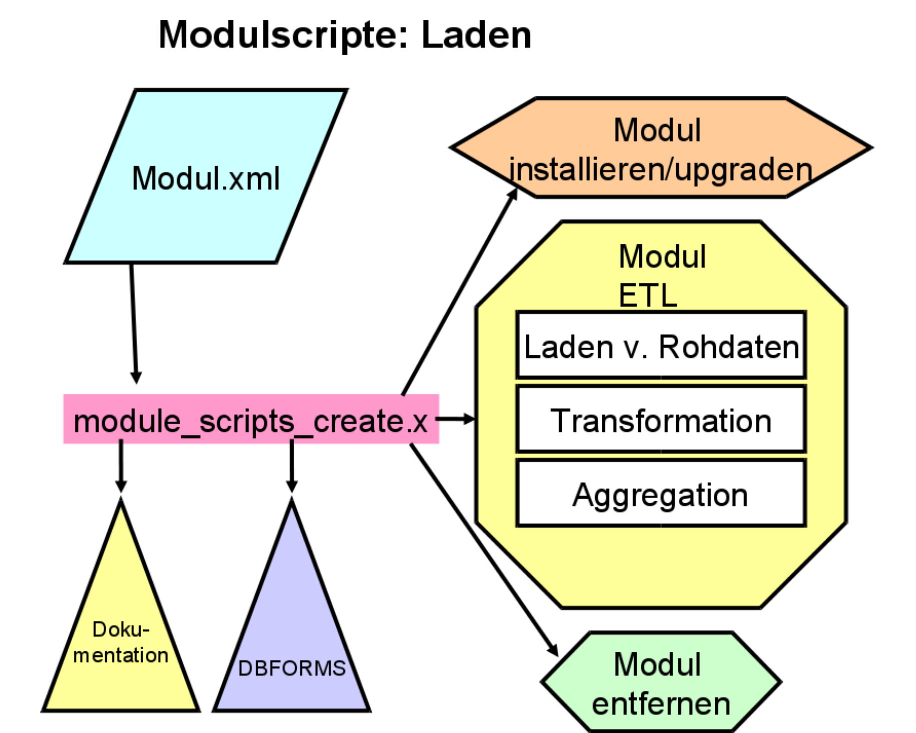
Die Ordnerstruktur eines Moduls spiegelt die Komponenten des Systems wieder. Es gibt je ein Verzeichnis für `datentabellen`, `schluesseltabellen` und `hilfstabellen`.

Die Installation eines Moduls ist in der Dokumentation des jeweiligen Moduls näher beschrieben. Module, die auf dem Kernmodul 2.1 oder höher basieren, haben einen einheitlichen Aufbau.

### 3.10.2Modulscripte im Kernmodul

Seit Version 2.1 werden die Datenbankschemata und Scripte der Module in einem einheitlichen Format zusammengestellt und in einer Datei `$SUPERX_DIR/db/module/$MODULNAME/conf/$MODULNAME.xml` gespeichert. Das XML-Format hat den Vorteil, dass die Scripte dynamisch für Postgres und Informix erzeugt werden können, und dass die Scripte vereinheitlicht werden. Aus dieser Datei werden die Scripte erzeugt, die das Modul jeweils für Postgres und Informix installieren / updaten /aktualisieren / überprüfen und entfernen. Die folgende Abbildung zeigt das Vorgehen:

Aus der xml-Datei werden die jeweiligen Scripte für die Installation, den Update, die Extraktions-, Transformations- und Ladescripte (ETL) und die Deinstallation erzeugt.




Die Modul-Scripte liegen als Shellscripte im Verzeichnis `$SUPERX_DIR/db/bin`, und sind an anderer Stelle im Detail erläutert.

| **module_install.x <<name>> <<pfad>>** | Installiert ein Modul, d.h. erzeugt die Tabellen, Views, Funktionen und Abfragen. Die Abfragen werden in den Themenbaum eingehängt. |
|---|---|
| **module_etl.x <<name>> <<pfad>>** | Aktualisiert das Modul, d.h. lädt die Rohdaten, Transformiert sie, und aggregiert die Hilfstabellen. Fehler werden in eine Log-Datei geschrieben, diese kann dann per Mail an einen Admin versandt werden. Wenn das Script erfolgreich durchläuft, wird die Tabelle `systeminfo` aktualisiert.<br>Das Script wird nicht direkt als Cronjob aufgerufen, sondern von einem Shellscript, das die Umgebungsvariablen und Parameter setzt, z.B. `cob_update.x`. |
| **module_drop.x <<name>> <<pfad>>** | Entfernt das Modul bzw. die Tabellen, Views, Funktionen und Abfragen. |
| **module_update.x <<name>> <<pfad>>** | Ein Modulupdate wird durchgeführt, d.h. eine neue Version des Moduls wird installiert (nicht zu verwechseln mit dem regelmäßigen Update im Sinne eines ETL-Prozesses). |
| **module_scripts_create.x <<name>> <<pfad>> <<Datenbanksystem(optional, INFORMIX, POSTGRES)>> <<Versionsnr.>>** | Erzeugt via XSL-Transformation die jeweiligen Scripte, die in den obigen Scripten aufgerufen werden, z.B. `bau_install_ids.sql` für das Installationsscript des Baumoduls beim Informix Dynamic Server. |

Neben den operativen Scripten erzeugt `module_scripts_create.x` auch html-Dateien zur Dokumentation eines Moduls in

`$SUPERX_DIR/db/module/<<Modulname>>/conf/<<Modulname>>.html`

(auch als rtf-Datei zu Einbindung in Modul-Dokumentationen) sowie zur Schnittstelle in

`$SUPERX_DIR/db/module/<<Modulname>>/rohdaten/<<Modulname>>_unload.html`

Darüber hinaus werden auch DBForms-Formulare erzeugt.

### 3.10.3Installation eines Moduls: Allgemeines Vorgehen

Das Vorgehen bei der Installation eines Moduls ist standardisiert.

1.Entpacken Sie das Modul in `$SUPERX_DIR`

2.Einrichtung der Umgebung in `SQL_ENV`; für jedes Modul sollten Beispielvariablen in `SQL_ENV_<<Modulname>>.sam` stehen. Bitte legen Sie hier auch Email-Adressen für log- und Fehlermails an.

3.Entladen der Rohdaten; auch hier müssen Umgebungsvariablen auf dem Quellrechner angepasst werden

(`<<Modulname>>_ENV`)

4.Kopieren der Rohdaten nach `<<Modulpfad>>/rohdaten`. Neuere SuperX-Module haben dafür vorgefertigte Scripte mit dem Namen `<<Modulname>>_copy.x`

5.Bei sehr großen Datenmengen bietet es sich an, die Rohdaten zunächst auf überschaubare kleine Dateien zu kürzen. Das Script
`<<Modulname>>_shrink.x`

kürzt alle "*_neu"-Tabellen auf 100 Zeilen. Sie können dies später mit <<Modulename>>_unshrink.x rückgängig machen.

6. Installieren Sie das Modul mit
   <<Modulname>>_erzeugen.x
7. Wenn die Installation erfolgreich war, können Sie das Modul aktualisieren mit
   <<Modulname>>_update.x (ggf. mit Parametern)
   d.h. die ETL-Prozesse werden gestartet (s.u.).
8. Wenn das Modul erfolgreich aktualisiert ist, wird eine Prüfprozedur gestartet, die die Daten plausibilisiert. Fehler und Warnungen finden Sie in der Datei $<<Modulname>>_ERRORDAT.

#### 3.10.3.1 Dateitransfer mit scp/rsync

Für den Transfer der Rohdaten wird in SuperX die dateibasierte Schnittstelle genutzt. Unter UNIX läßt sich dieser Transfer vollends automatisieren, indem die Programme scp oder rsync auf der Basis des OpenSSH-Pakets genutzt werden[22] . Beide setzen auf das ssh-Protokoll 2 auf und stellen somit einen verschlüsselten Dateitransfer sicher.

In den jeweiligen Modulen wird im Verzeichnis rohdaten eine Beispieldatei mit dem Namen <<MODULNAME>>_ENV.sam ausgeliefert, die Sie umbenennen können nach <<MODULNAME>>_ENV. Darin werden am Ende der Datei die Parameter zum Kopieren festgelegt, also die Userkennung REMOTE_USER, der Hostname REMOTE_HOST, und die Methode des Kopierens (COPY_METHOD) sowie die jeweiligen Zielpfade. Diese Umgebungsvariablen werden von dem jeweiligen Script <<modulname>>_copy.x benutzt.

Damit die Passworteingabe entfällt, muss man wie folgt vorgehen:

Loggen Sie sich zunächst testweise einmal ein. Wenn Sie z.B. vom COB-Server auf den SuperX-Server kopieren wollen, loggen Sie sich als user cob auf cobhost ein mit

```
ssh superx@superxhost
```

Beim ersten Mal müssen Sie die Sicherheitsabfrage mit "yes" bestätigen.

Erzeugen Sie auf dem Quellrechner einen öffentlichen Schlüssel mittels ssh-keygen -t rsa, wobei man eine leere Passphrase vergibt (Achtung: mögliche Sicherheitslücke!). Der öffentliche Teil dieses Schlüssels (~/.ssh/id_rsa.pub) muss auf dem Zielrechner in die Datei ~/.ssh/authorized_keys eingefügt werden, ggf. muss die Datei neu erzeugt werden.

Wenn z.B. auf dem COB-Server unter der Kennung cob ein Key wie folgt erzeugt wurde:

**Beispieleintrag eines Public Keys**

```
ssh-rsa AAAAB3Nza…[hier viele kryptische Zeichen]
                                       …pg6VkCc= cob@cobhost
```

Dann wird genau diese Zeile in der Datei /home/superx/.ssh/authorized_keys angefügt (die Datei kann mehrere PublicKeys enthalten, ein Eintrag pro Absatz).

[22] Details zu rsync siehe Dr. Boris Pasternak, Dr. Uwe Meyer-Gruhl (2003). Der Gleich-Macher. Dateien mit Rsync synchronisieren. c't 10/2003,S. 116ff.

Danach sollte z.B. der Login vom `cobhost` als user `cob` mit `ssh superx@superxhost` ohne Passworteingabe klappen. Wenn nicht, schalten Sie das Logging mit `ssh -v superx@superxhost` ein. Eine Möglichkeit ist, dass die PublicKey-Authentifizierung in der Konfigurationsdatei des SSHD (normal `/etc/ssh/ssh_config`) abgeschaltet ist.

Sie können außerdem noch einschränken, von welchem Host die obige Authentifizierung ermöglicht wird. Dazu setzen Sie den Parameter `"from=*.uni-xy.de"` davor, z.B.

| **Einschränkung "from" in authorized_keys** | `from="*.uni-xy.de" ssh-rsa AAAAB3Nza…[hier viele kryptische Zeichen]…pg6VkCc= cob@cobhost` |
|---|---|

Wenn Sie die Kopiermethode `scp` benutzen, und die obige "authorized_keys"-Metohde mit PublicKey nicht nutzen wollen, können Sie auch mit Private Keys arbeiten (siehe SSH-Doku). Dazu können sie in der *_ENV-Datei in dem Parameter SCP_OPTS den Verweis auf den private Key setzen.

| **SCP_OPTS in *_ENV: Beispiel COB_ENV** | `SCP_OPTS="-p -B -i /home/cob/.ssh/superx_key"`<br>`export SCP_OPTS` |
|---|---|

### 3.10.3.2SuperX-Java-Client zum Entladen von Quell-Datenbanken

Zum Entladen aus dem operativen Vorsystem wird unter Informix `dbaccess` genutzt. Unter Postgres wird generell der SuperX-JAVA-Client zum Entladen genutzt, denn SuperX benötigt ein spezielles, an Informix angepasstes CSV-Format, das sich mit Bordmitteln von Postgres (copy-Befehl) nicht erzeugen lässt. Es kann aber auch sinnvoll sein, aus der Informix-Datenbank mit SuperX-JAVA-Client zu entladen, z.B. wenn Sie kein UNIX-dbaccess auf dem Vorsystem installiert haben.

Wenn Sie das jew. operative Vorsystem im PUSH-Verfahren entladen wollen, d.h. die Rohdaten werden auf dem Vorsystem entladen und auf den SuperX-Rechner kopiert, dann müssen Sie spezielle Vorkehrungen treffen. SuperX nutzt generell zum Entladen eigene Java-Klassen. Beim Entladen im PULL-Verfahren sind diese Klassen vorhanden, denn die Entladeroutine läuft auf dem SuperX Rechner. Wenn Sie aber PUSH nutzen wollen, werdendie SuperX-Java-Klassen auf dem Liefersystem benötigt, und die Entladeroutine muss konfiguriert sein. Im Folgenden nutzen wir das Beispiel "Entladen im Push-Verfahren aus SVA-GX unter Postgres". Gehen Sie dazu wie folgt vor:

- Kopieren Sie die Dateien mit der Endung `*.jar` vom SuperX-Rechner im Verzeichnis `$SUPERX_DIR/tomcat/webapps/superx/WEB-INF/lib` auf den Quellrechner in ein Unterverzeichnis `lib` unter `rohdaten` (z.B. `/home/sva/superx/rohdaten/lib`). In `rohdaten` liegt die bisherige Entladeroutine (z.B. `sva_unload.x`).
- Fügen Sie dann folgenden Passus aus der Datei `$SUPERX_DIR/db/bin/SQL_ENV` in die Umgebungs-Datei der Entladeroutine, z.B. `SVA_ENV`:

```
#Pfad zu den SuperX-Java-Libraries
LIB_PATH=/home/sva/superx/rohdaten/lib
TOMCAT_LIB=LIB_PATH
#Der JDBC_CLASSPATH enthält alles, was der jdbc-Client in superx für den Da-
tenbankzugriff braucht.
JDBC_CLASSPATH=$TOMCAT_LIB/pg74.214.jdbc3.jar:$TOMCAT_LIB/ifxjdbc.jar:
$LIB_PATH/superx4.0.jar:$TOMCAT_LIB/commons-lang-2.0.jar:$TOMCAT_LIB/xalan2-
```

```
6-0.jar:$LIB_PATH/ant.jar:$TOMCAT_LIB/jfor-0.7.2rc1.jar:$TOMCAT_LIB/httpu-
nit.jar:$TOMCAT_LIB/nekohtml-0.9.3.jar:$LIB_PATH/freemarker.jar
export JDBC_CLASSPATH
#Der XML-Classpath enthält alle Libraries für XML-Tools in SuperX
XML_CLASSPATH=$TOMCAT_LIB/xmlParserAPIs.jar:$TOMCAT_LIB/xercesImpl-
2.7.0.jar:$TOMCAT_LIB/avalon-framework-cvs-20020806.jar:$TOMCAT_LIB/bati-
k-all-1.7.jar:$TOMCAT_LIB/logkit-1.0.1.jar
export XML_CLASSPATH
```

Wir "missbrauchen" also die (nur intern genutzte) Variable TOMCAT_LIB, die auf SuperX-Seite zu den benötigten Java-Bibliotheken zeigt.

Wenn dann noch die Variablen DB_PROPERTIES und LOGGER_PROPERTIES korrekt gesetzt sind, kann die Entladeroutine bei SX_CLIENT=jdbc (Wenn Sie unter Windows entladen, oder Informix ohne dbaccess entladen wollen) oder SX_CLIENT=psql (wenn Sie Postgres unter UNIX nutzen) mit Java entladen.

#### 3.10.3.3Modulupdate in mandantenfähigen Installationen

Der Modulupdate in mandantenfähigen Installtion findet in einer SuperX-Installation statt, allerdings werden die einzelnen Scripte mit unterschiedlichen Umgebungsvariablen, wie sie in SQL_ENV.<<MANDANTID>> definiert ist, z.B. SQL_ENV.FHRO.

In der SQL_ENV.<<MANDANTID>> werden unterschiedliche Pfade für den jeweiligen *_LOAD_PFAD gesetzt, wobei in der Regel die Mandandid ein Unterverzeichnis vom "normalen" LOAD_PFAD ist. So ist z.B. beim COB-Modul folgender Pfad anzusetzen:

Normale SuperX-Installation:

COB_LOAD_PFAD=$SUPERX_DIR/db/module/cob/rohdaten

Mandantenfähige SuperX-Installation:

COB_LOAD_PFAD=$SUPERX_DIR/db/module/cob/rohdaten/FHRO

Unterhalb von FHRO befindet sich noch einmal die Entladeroutine sowie das Unterverzeichnis unl mit den Rohdaten. Dieses Verzeichnis FHRO kann der Einfahheit halber auch ein symbolischer Link auf den gemounteten COB-Rechner sein.

Durch Setzen der Mandantennummer in der Umgebungsvariable MANDANTID in der jeweiligen SQL_ENV des Mandanten werden die ETL-Scripte anders ausgeführt: Die Logdateien werden jeweils mit der Mandantennummer versehen (z.B. L_cob_updateFHRO.log), damit die Übersicht nicht verloren geht und der gleichzeitige Update mehrerer Mandanten in eine rsuperX-Installation möglich ist.

Außerdem können weitreichende Steuerungsmechanismen im Modulupdate eingesetzt werden: Nach jedem ETL-Schritt können optional mandantenspezifische Scripte aufgerufen werden. Diese müssen folgende Namenskonvention einhalten:

<<Scriptname>>_<<MANDANTID>>.sql

Also für eine hochschulspezifische Transformation im COB-Modul des Mandanten FHRO wird eine Datei namens

cob_trans_FHRO.sql

mit entsprechenden SQL-Anweisungen angelegt.

#### 3.10.3.4 Upgrade eines Moduls: Allgemeines Vorgehen

Zum Upgrade bzw. zum Zurücksetzen des Moduls auf den Auslieferungszustand entpacken Sie das Paket in $SUPERX_DIR und gehen in das Verzeichnis $SUPERX_DIR/db/module/<<Modulname>>/upgrade und führen das Script aus:

<<Modulname>>_upgrade.x

Ausnahme: beim Kernmodul gibt es i.d.R. ein spezielles Upgrade Script.

Die Logdatei lautet upgrade.log, im Mandantenfähigen Betrieb "upgrade<<MANDANTID>>.log".

Wenn Sie einen separaten Tomcat-Rechner betreiben, müssen Sie das Paket dort ebenfalls entpacken, und vom Datenbankserver die Datei $SUPERX_DIR/webserver/tomcat/webapps/superx/WEB-INF/dbforms-config.xml an die gleiche Stelle auf den Tomcat Rechner kopieren. Ein nochmaliges Ausführen des Upgrade Scriptes ist nicht nötig, weil dies nur die Datenbank betrifft.

#### 3.10.3.5 Entfernen eines Moduls

Wenn Sie ein Modul nicht mehr benötigen, starten Sie das Script
$SUPERX_DIR/db/module/<<Modulname>>/<<Modulname>>_modul_entfernen.x.
Dieses Script löscht alle Tabellen, Prozeduren und Abfragen aus der Datenbank, und löscht auch die Einträge im Themenbaum. Danach können Sie den Pfad $SUPERX_DIR/db/module/<<Modulname>>löschen.

Wenn Sie nur die Inhalte der Daten- und Hilfstabellen des Moduls löschen wollen (z.B. aus Datenschutzgründen), ohne das ganze Modul zu deinstallieren, können Sie dies mit folgendem Befehl tun:

DOSQL $SUPERX_DIR/db/module/<<Modulname>>/<<Modulname>>_purge_pg.sql (für Postgres)

bzw.

DOSQL $SUPERX_DIR/db/module//<<Modulname>>/<<Modulname>>_purge_ids.sql (für Informix)

## 3.11 Überwachung und Performance

SuperX besteht aus verschiedenen Komponenten, die jeweils eigene Überwachungsmerkmale und Performance-Mechanismen besitzen.

### 3.11.1 Überwachung und Performance der Webanwendung

Die Webanwendung baiert auf Tomcat, und die Logdateien des Tomcat liegen standardmäßig im Verzeichnis $SUPERX_DIR/webserver/tomcat/logs. Die Logdateien im Einzelnen:

- Logging von Tomcat: catalina.out bzw. localhost.xxx.out
- Logging der SuperX-Webanwendung jeweils in superx_default.log (statt "default" ggf. die Mandantenid) für allgemeines SQL-Logging, und superx_default_xml.log für das Logging der XML-Ausgabe des XML-Frontends.
- dbforms.log für Logging der DBForms-Komponente

Alle Logging-Ausgaben lassen sich flexibel an verschiedenen Stellen steuern:

- Das Tomcat-Logging lässt sich in der Datei `$SUPERX_DIR/webserver/tomcat/common/classes/log4j.properties` steuern
- Das Ausmaß des Loggings der SuperX-Webanwendung: Im propadmin wird der Logging-Level für die SQL-Ausgabe sowie für die XML-Ausgabe festgelegt.
- Das Logging für DBFORMS wird in der Datei `$SUPERX_DIR/webserver/tomcat/webapps/superx/WEB-INF/log4j.properties` festgelegt.
- Das Logging der java-bezogenen SuperX-DB-Anwendung wird in der Datei `$SUPERX_DIR/db/conf/logging.properties` gesteuert.

Die SQL-Scripte der SuperX-Abfragen können in der Java-Konsole des Applets sowie in der o.g. superx_default.log eingesehen werden. Bitte beachten Sie dabei, dass bei SQL-Fehlern nur im Entwicklungsmodus die genaue Stelle des Auftretens ermittelt werden kann.

#### 3.11.1.1Steuerung des SQL-Logging im SuperXManager

Bei der Auslieferung von superX werden alle SQL-Befehle einzeln an den Server übertragen und ausgeführt, um im Falle von Fehlern schnell die Position ermitteln zu können. Dieser Modus nennt sich "Entwicklungsmodus" und ist im propadmin die Voreinstellung, kann aber an dieser Stelle auch geändert werden. Die Änderung wäre nach dem Start des Tomcat aktiv.

Einige Parameter des propadmin lassen sich auch im laufenden Betrieb umstellen. Im SuperX-Manager-Servlet können Sie z.B. den Entwicklungmodus umschalten.

Gehen Sie dazu im XML-Frontend auf ein Bearbeitungsformular von Benutzerrechten und klicken Sie oben rechts auf den Link **SuperX-Manager**
Der Schalter **Entwicklungsmodus** an/aus lässt sich auch im laufenden Betrieb umstellen. Klicken Sie dazu auf an/aus und dann auf "Übernehmen". Auch die **Tomcat-logs** lassen sich hier leeren. Unten zeigt der Manager jeweils den letzten SQL vor bzw. nach Freemarker-Transformation an, sowie den letzten XML-Strom.

In Produktionsumgebungen empfpehlen wir, den Entwicklungsmodus abzuschalten, da die Abfragen dann 25-50% schneller laufen.

### 3.11.1.2Java-Monitoring mit JConsole

Mit Java 1.6 und Tomcat 5.5 gibt es eine komfortable Möglichkeit, den Server zu überwachen. Vor dem Start von Tomcat setzen Sie die Option CATALINA_OPTS wie folgt:

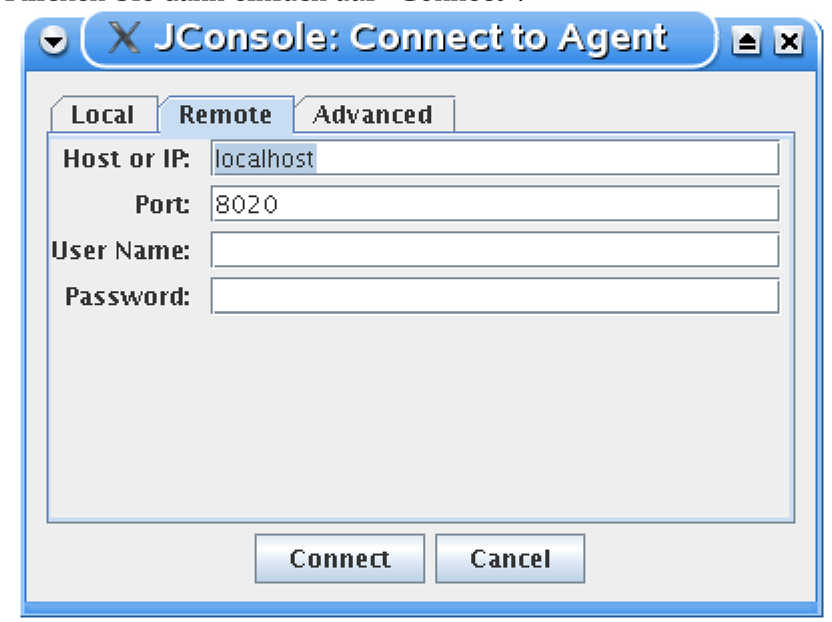
**Achtung: Alle Zeilen in eine Zeile tippen, die Umbrüche kommen nur durch das Layout**

```
CATALINA_OPTS="-Dcom.sun.management.jmxremote -Dcom.sun.mana-
gement.jmxremote.port=8020
-Dcom.sun.management.jmxremote.ssl=false -Dcom.sun.manage-
ment.jmxremote.authenticate=false"
export CATLINA_OPTS
```

Sie starten den Tomcat dann mit einer Überwachungsschnittstelle auf Port 8020, die Sie dann von einem (entfernten) Client auswerten können:

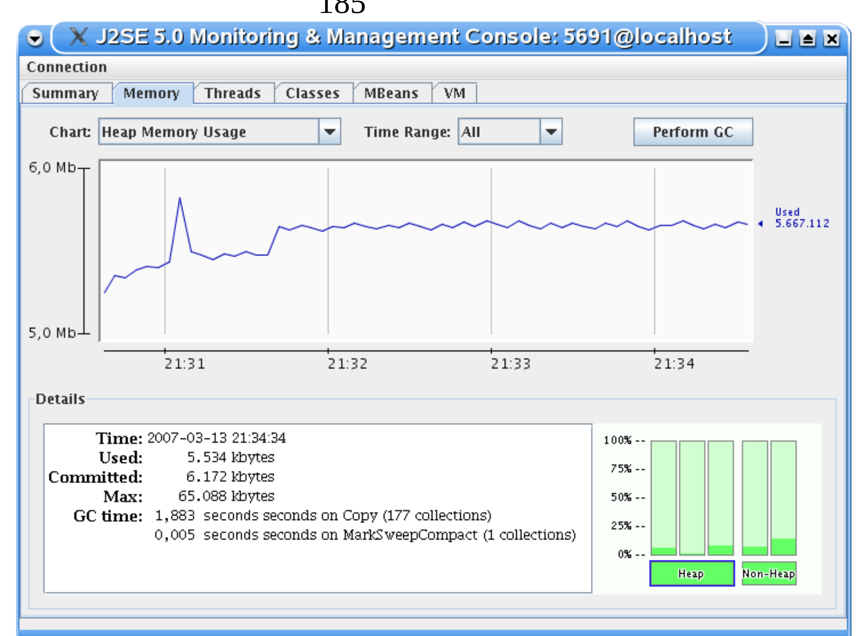
**Starten Sie das Programm jconsole Bei einem entfernten Rechner geben Sie den Rechnernamen und Port an**

jconsole
Klicken Sie dann einfach auf "Connect".




Diese Anwendung liefert detailliert Aufschluss über den Server:

**Hier sehen Sie die Arbeitsspeicher-Auslastung des Tomcat Servers.**




Wir empfehlen, im Produktivbetrieb dies abzuschalten (Sicherheitslücke und Performance-Kosten). Eine detailliertere Anleitung finden Sie hier:
http://blog.linkwerk.com/entry/cl/2007-05-08T12.00.00

Generell empfehlen wir, den Tomcat im Produktivbetrieb jede Nacht einmal neu hochzufahren, im SuperX-Kernmodul wird dazu ein Beispielscript ausgeliefert (`db/bin/restart_tomcat.x`). Ein weiteres nützliches Script prüft z.B. alle 5 Minuten, ob der Server noch läuft; wenn nicht dann wird er automatisch hochgefahren (`db/bin/check_restart_tomcat.x`).

## 3.12Downloads einrichten und verteilen

SuperX bietet die Möglichkeit, beliebige Dateien über die Webapplikation an Anwender auszuliefern, z.B. um einen Downloadbereich einzurichten[23]. Die Downloads können einzelnen Usern oder Gruppen sowie Institutionen und Themen zugeordnet werden.

### 3.12.1Konfiguration

Die Download-Dateien werden in dem geschützten Verzeichnis der Webapplikation gespeichert. Um die Dateien gezielt in einem Verzeichnis zu speichern, muss man ggf. das Attribut "directory" des Feldes "datei" in der Tabelle sx_downloads in der Datei `dbforms-config.xml` setzen, standardmäßig ist dies (relativ zu dem Startpfad von Tomcat) `"../webapps/superx/WEB-INF/downloads"`.

Mit dem Attribut "encoding" (default "false") wird festgelegt, ob der Dateiname vom Original übernommen werden soll ("false") oder ob eine eindeutige Zufalls-Zeichenkette ("true") erzeugt werden soll. Die Endung der Datei wird bei letzterem beibehalten.

[23] Achtung: diese Funktion ist bei mandantenfähigen Installationen nicht ohne weiteres nutzbar, hier sind Anpassungen notwendig. Wenden Sie sich bei Bedarf an den SuperX-Support.

Gleichzeitig werden der Dateiname und diverse andere Metadaten in der Tabelle `sx_downloads` gespeichert. Wenn ein Anwender einen Download abruft, dann wird die Datei im SuperX-Servlet geladen und über http(s) ausgeliefert.

Die Auslieferung von Dateien wird defaultmäßig **protokolliert** und kann über die Maske "Downloadstatistik" abgerufen werden. Sie können diese Funktionalität (z.B. aus Datenschutzgründen) sperren, indem Sie die Konstante "`DOWNLOAD_PROTOKOLL`" statt auf "1" auf "0" setzen - damit werden keine Download-Aktivitäten in SuperX protokolliert (was aber nicht bedeutet, dass dies auch im Webserver-Log nicht mehr passiert, die dortige Protokollierung sowie die Tomcat-eigene Protokollierung ist davon unabhängig).

Außerdem können Sie die maximale Größe von Dateien festlegen. Dafür gibt es in der `web.xml` einen Parameter "maxUploadSize", der die maximal Größe (in Bytes) beschreibt:

```
<!--=========== DbForms Controller Servlet ==============-->
    <servlet>
      <servlet-name>control</servlet-name>
      <servlet-class>org.dbforms.servlets.Controller</servlet-class>

      <init-param>
        <param-name>maxUploadSize</param-name>
        <param-value>800000</param-value>
      </init-param>
    </servlet>
```

### 3.12.2Tabellenstruktur

Es gibt eine Tabelle `sx_downloads` mit folgenden Feldern:

| Feldname | Feldtyp | Größe | Default | Not Null | Beschreibung |
|---|---|---|---|---|---|
| tid | SERIAL | 4 | | true | Primärschlüssel |
| name | CHAR | 255 | | false | Titel |
| ch110_institut | CHAR | 10 | | false | Kostenstelle/Institut |
| bezugsdatum | DATE | 4 | | false | (für Ermittlung Bezugsjahr,- Monat oder Sem.) |
| importdatum | DATE | 2 | | false | Datum des Imports in die SuperX-Datenbank |
| kommentar | TEXT | 32000 | | false | Kommentar für Website (Datenlegende o.ä.). |
| kommentar_www | CHAR | 255 | | false | Verweis auf andere Website für längere und gelayoutete Kommentare oder Dokumentationen. |
| contenttype | CHAR | 50 | | false | Mime-Type der Datei (pdf, html etc). |
| datei | CHAR | 255 | | true | Pfad zum geschützten Verzeichnis (relativ zu `$SUPERX_DIR/webserver/tomcat/webapps/superx/WEB-INF/downloads`) |
| gueltig_seit | DATE | 2 | | false | Soll Download angezeigt werden von … |
| gueltig_bis | DATE | 2 | | false | Soll Download angezeigt werden bis… |

Desweiteren gibt es eine Tabelle `sx_keywords` zur Erhebung der Stichworte:

| Feldname | Feldtyp | Größe | Default | Not Null | Beschreibung |
|---|---|---|---|---|---|
| tid | SERIAL | 4 | | false | Tupelidentifier |

| | | | | | |
|---|---|---|---|---|---|
| name | CHAR | 255 | | false | Stichwort |
| parent | INTEGER | 4 | | false | Übergeordnetes Stichwort<br>Wird derzeit noch nicht ausgewertet. |

Die Zuordnung zwischen Download und Stichwort findet in der Tabelle `download_keyw_bez` statt:

| Feldname | Feldtyp | Größe | Default | Not Null | Beschreibung |
|---|---|---|---|---|---|
| keyword_id | INTEGER | 4 | | false | |
| download_id | INTEGER | 4 | | false | |

### 3.12.3Berechtigung für Downloads

Die Berechtigungen für die Downloads werden über die SuperX-Gruppen- bzw. Userrechte verwaltet. Dazu werden eigene Tabellen `user_download_bez` und `group_download_bez` erzeugt, für die auch Pflegeformulare existieren. Die Institutions-Berechtigung wird auch Bordmitteln von SuperX realisiert, d.h. die Anwender erhalten über ihre Zuordnung zur jeweiligen Kostenstelle in der Tabelle `user_institution` das Recht für die Kostenstelle und alle jeweils untergeordneten Kostenstellen.

Einzelne vorgefertigte Masken sind bereits eingerichtet und werden im Folgenden beschrieben.

### 3.12.4Masken zur Erzeugung und Verteilung von Downloads

Im XML-Frontend finden Sie die Download-Masken im Themenbaum-Ast "Administration".

#### 3.12.4.1Download suchen

Mit der Maske "Download suchen" können sie einzelne Downloads einrichten, bearbeiten oder löschen.

In der Suchmaske können Sie verschiedene Parameter einschränken. Wenn ein Stichwort oder eine Kostenstelle ausgewählt wird, dann werden alle Downloads mit diesem oder untergeordnetem Stichwort/ Kostenstelle gefunden.

**Download suchen**

Bitte schränken Sie Ihre Auswahl ein:

Stichwort
Institution
Jahr
Monat
Suchwort

Abschicken
Zurücksetzen

Das Freitext-Feld **Suchwort** bezieht sich auf den Namen des Downloads.

Die Ergebnistabelle zeigt die Downloads. Wenn Sie als Administrator gekennzeichnet sind (Feld `administration` in `userinfo` steht auf "1"), dann können Sie die Downloads nicht nur laden, sondern auch bearbeiten sowie zu Usern/Gruppen bzw. Themen zuordnen.

**Download suchen**

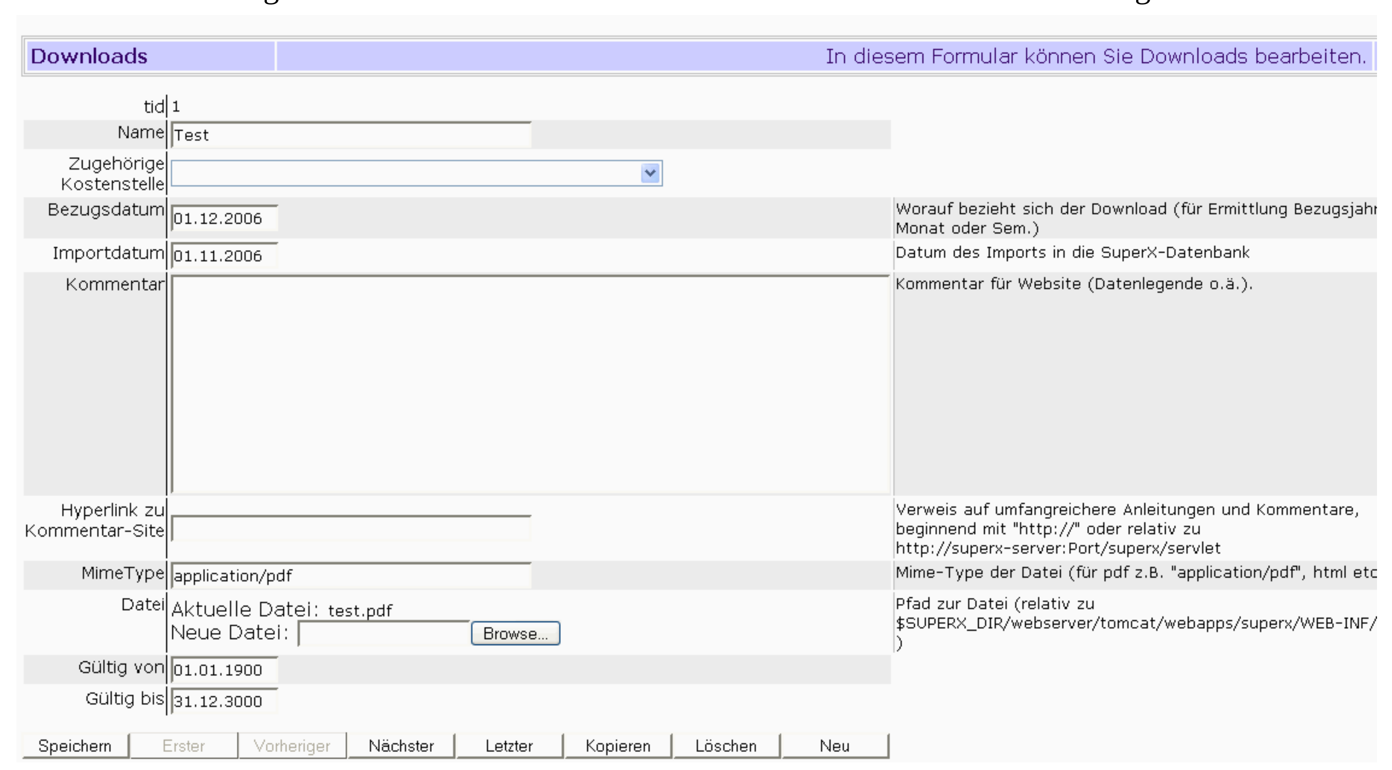
Institution: **FB ABV gesamt** ; Stand: 01.01.2003

| Name | Kommentar | Kommentar (www) | Laden | Bearbeiten | User- und Gruppenrechte | Stichworte zuordnen |
|---|---|---|---|---|---|---|
| Test | | | | | | |

Datensatz 1 - 1 von insgesamt 1 Satz.

### 3.12.4.2 Download bearbeiten: Metadaten und Dateien

In der Bearbeitungsmaske erscheinen die oben beschriebenen Felder nebst Erläuterungen.




Sie können, müssen aber nicht, einem Download einer einzelnen Kostenstelle zuordnen. Hierarchische Anordnungen werden dabei suchbar, d.h. wenn ein Anwender in der Insitutions-Sicht des Organigramms eine Kostenstelle auswählt, dann werden alle Downloads mit untergeordneten Kostenstellen ebenfalls gefunden.

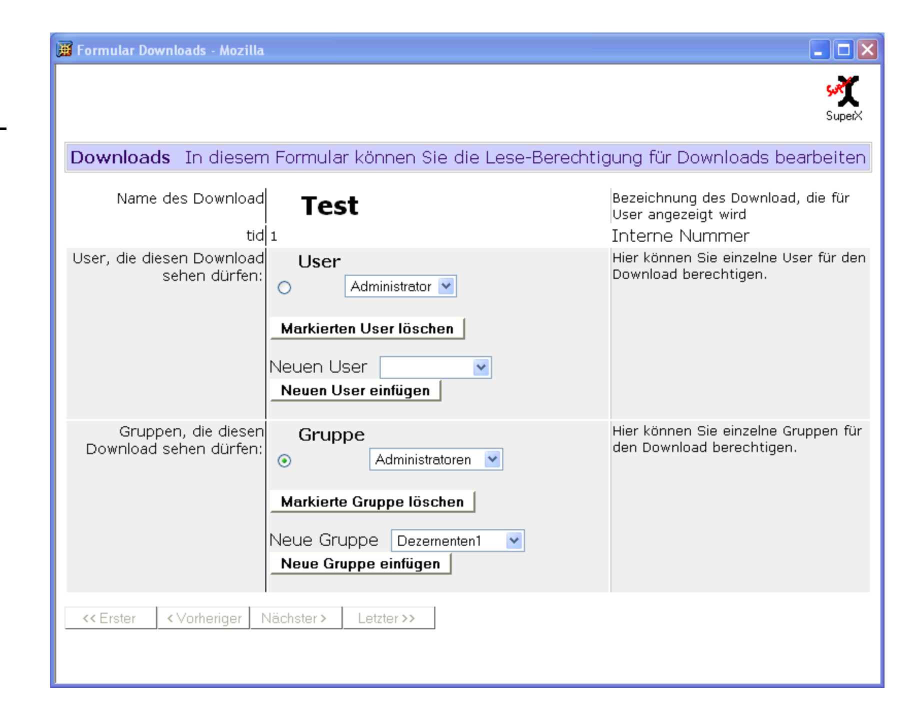
Sie können Dateien Hochladen, inden Sie in der Zeile Datei eine neue Daten festlegen. Ansonsten wird darüber der aktuelle Dateiname festgelegt. Wichtig ist, dass der Dateiname in dem Verzeichnis `$SUPERX_DIR/webserver/tomcat/webapps/superx/WEB-INF/downloads` eindeutig ist. Außerdem funktioniert der Browser-basierte Upload nur mit kleinen Dateien, größere Dateien sollten Sie manuell in das Verzeichnis `$SUPERX_DIR/webserver/tomcat/webapps/superx/WEB-INF/downloads` kopieren.

Sie können auch Datensätze kopieren, allerdings werden nur die Metadaten werden kopiert, sie müssen dann eine neue Datei hochladen.

### 3.12.4.3User- und Gruppenrechte auf Downloads

Mit der Schaltfläche unter "User- und Gruppenrechte" sehen Sie das Bearbeitungsformular.

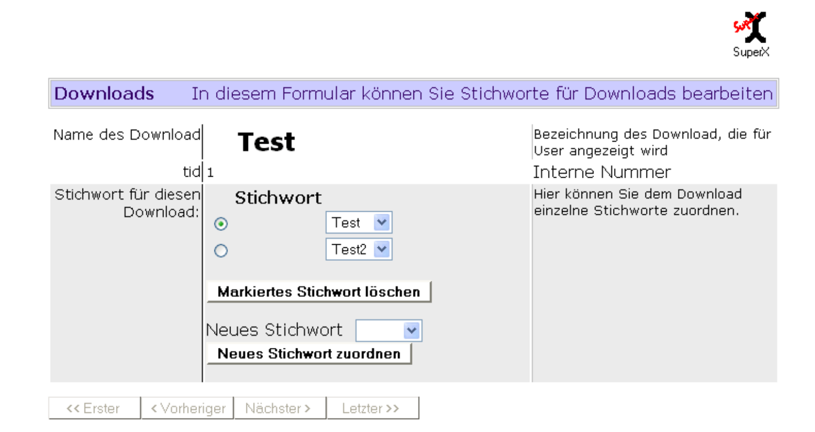
Sie können jeweils einzelne User oder Gruppen zuordnen, die Funktionalität entspricht der Berechtigung für Sichten.




### 3.12.4.4Stichworte für Downloads

Mit der Schaltfläche unter "Stichworte zuordnen" sehen Sie das Bearbeitungsformular.

Sie können jeweils ein oder mehrere Themen zuordnen.




## 3.13Migrationsprojekte

Es gibt verschiedene Szenarien zur Migration von SuperX, hierzu werden Empfehlungen gegeben.

### 3.13.1Postgres: Wechsel auf der Zeichencodierung auf UTF-8

Unter Postgres bietet es sich an, von Postgres ISO-Codierung zu Postgres-UTF-8 Codierung zu wechseln. Für die Umsetzung der Zeichencodierung von Dateien gibt es in Postgres eingebaute Unterstützung: Wenn ein Text-Dump einer ISO-Datenbank erzeugt wird, dann steht im Kopf der Datei die Encodierung. Wenn man diese Datei mit psql in eine UTF-8-DB füttert, wird automatisch von ISO nach UTF-8 konvertiert. Für die Umcodierung der Datenbank hat sich also folgendes Vorgehen bewährt:

Exportieren Sie die ISO-Datenbank mit pg_dump im Format "plain text". Es wird eine sql-Datei erzeugt, im folgenden Beispiel für die Datenbank mit dem Namen "$DBNAME":

```
pg_dump -f $DBNAME.sql  $DBNAME
```

Falls Sie direkt beim Dump eine zip-Datei erzeugen wollen, nutzen Sie folgendes Script:

```
pg_dump $DBNAME | gzip >$DBNAME.sql.gz  2>dump.err
```

Wechsel Sie dann in eine Postgres-Installation, die UTF-8 unterstützt, und erzeugen Sie die Datenbank neu:

```
createdb –encoding=UTF-8 $DBNAME
```

Dann importieren die Datenbank mit:

```
psql $DBNAME < $DBNAME.sql
```

### 3.13.2Migration von Postgres zu Informix

Eine direkte Konvertierung von Postgres nach Informix geht nicht, wg. der stored procedures. Wir raten zu folgendem Vorgehen:

- Entpacken Sie die jeweils genutzten Release-Module für Informix  in $SUPERX_DIR. Die Versionsnummern sollten exakt den Postgres-Modulen entsprechen, die Sie bereits nutzen.
- Kopieren Sie die Datei $SUPERX_DIR/db/bin/SQL_ENV nach SQL_ENV.pg
- Ändern Sie die Datei $SUPERX_DIR/db/bin/SQL_ENV, indem Sie dort die Umgebungsvariablen DATABASE und SX_CLIENT nach INFORMIX / dbaccess ändern. INFORMIXSERVER etc. müssen Sie ebenfalls eintragen
- Kopieren Sie die Datei
  $SUPERX_DIR/webserver/tomcat/webapps/superx/WEB-INF/db.properties
  nach
  db_pg.properties, und legen Sie die Umgebungsvariable DB_PROPERTIES in der Datei
  $SUPERX_DIR/db/bin/SQL_ENV.pg auf diese Datei
- Laden Sie die Informix-Umgebung mit
  . $SUPERX_DIR/db/bin/SQL_ENV
- Installieren Sie das Informix-Kernmodul in der shell in
- $SUPERX_DIR/db/install
  mit
  kernmodul_erzeugen.x
- Passen Sie die jetzige Datei $SUPERX_DIR/webserver/tomcat/webapps/superx/WEB-INF/db.properties so an, daß der INFORMIXSERVER gefunden wird. Dafür haben wir ein Tool namens "propadmin.x", mit dem Sie die DB-Verbindung eintragen und testen können. Wenn Sie keine graphische Umgebung haben, müssen Sie die Informix-Parameter in der Datei manuell anpassen, Musterdateien liegen im gleichen Verzeichnis.

- Installieren Sie die jew. Module mit dem *erzeugen.x-Script
  (ohne jeweils den Update zu starten)
- Entladen Sie die Postgres-Module, indem Sie in das jew. Modulverzeichnis gehen, und
- Laden Sie die Postgres-Umgebung mit
  . $SUPERX_DIR/db/bin/SQL_ENV.pg
- Entladen sie das Modul mit <<Modulname>>_backup.x
- Es werden csv-Dateien in das Unterverzeichnis tmp geschrieben
- -Laden Sie die Informix-Umgebung mit
  . $SUPERX_DIR/db/bin/SQL_ENV
- Laden sie das Modul mit <<Modulname>>_restore.x

Voila! Sogar die Logins bleiben erhalten.

Probleme sind nur an folgenden Stellen zu erwarten:

- Die Kernmodul-Tabellen maskeninfo und felderinfo könnten beim CSV-Upload Probleme mit Umbrüchen machen. Mit dem jew. Script <<Modul>>_masken_einspielen_ids.x kann man die Masken aber manuell laden.
- Die Tabelle des SOS-Moduls lehr_stg_ab enthält Spalten mit "not null"-Constraints. Beim CSV-Upload werden Leerstrings aber NULLs, so daß die Tabelle sich nicht laden läßt. Man kann da über eine temp.Tabelle gehen
- Hochschul-spezifische Tabellen und Masken müssen manuell geladen werden. Ggf. müssen wir Anpassungen für Postgres-Syntax machen.

### 3.13.3Migration von SuperX zu HISinOne / Edustore

In HISinOne / Edustore ist folgendes fest vorgegeben:

- Das DBMS ist Postgres
- Die Zeichencodierung ist UTF-8

Vor einer Migration zu HISinOne / Edustore müssen Sie also o.g. zuerst erledigen. Eine Anleitung finden Sie oben.

- Wenn die Migration zu Postgres / UTF-8 gelungen ist, müssen Sie wie folgt vorgehen:
- Entpacken Sie das HISinOne Release auf dem Server, und richten Sie den Qisserver ein (databases.xml etc).
- Bei der Neuinstallation von HISinOne müssen Sie leere Postgres-Datenbanken anlegen. Sie legen die eduata- und edugeta-Datenbank an. Für die eduetl-Datenbank ist dies in Ihrem Falle nicht notwendig. Sie verlinken die oben migrierte Datenbank in der databases.xml
- Sie starten in der BI-Administration den Upgrade der jeweiligen Module, beginnend mit dem Kernmodul. Dies können Sie über den Browser realisieren, oder über Shell- bzw. ANT Scripte

# 4Bestandteile des Kernmoduls: Die Referenz

Das Kernmodul besteht aus der Userverwaltung, der Maskenverwaltung sowie aus dem Organigramm. Die wichtigsten Tabellen des Kernmoduls sind im folgenden aufgeführt.

Die Basisdaten und zusätzlichen Schlüssel der weiteren Module werden nach folgender Konvention nach SuperX übernommen:

| Tabellenname | <Basisdatenbank>_<Name der Tabelle in der Basisdatenbank> |
|---|---|
| Beispiele | "cob_busa" oder "mbs_inst" |

## 4.1Die Userverwaltung

Die User- und Gruppenverwaltung geschieht über eine Reihe von Tabellen, die User, Gruppen, Institutionen, thematische Sachgebiete und einzelne Abfragen in Beziehung setzen. Die Tabellen werden im folgenden dokumentiert. Client-Formulare zur Benutzerverwaltung finden Sie .

### 4.1.1Verwaltung einzelner User

Die Userverwaltung beruht auf einige Tabellen, die Stammdaten und Beziehungen zu anderen Tabellen (z.B. Masken) abbilden.

#### 4.1.1.1Tabelle userinfo

Die Tabelle `userinfo` enthält die Angaben zur Person, d.h. email, Nutzerkennung, Passwort etc.

**Tabelle userinfo**

| **Feld** | **Erläuterung** | **Beispiel** |
|---|---|---|
| tid | Id-Nummer | 1 |
| benutzer | Nutzerkennung für Anmeldung | testuser |
| kennwort | Passwort alter Client | frsgrgr |
| name | Name der Person | Ein Testuser ohne Name |
| max_versuch | Maximale Logins | 5 |
| akt_versuch | Aktuelle Login-Versuche | 0 |
| email | Email-Adresse | test@uni.de |
| administrator | Administrator-Rechte (1=ja, 0=nein) | 0 |
| archiv_recht | Leserechte auf Archiv (1=ja, 0=nein) [im alten Client] | 0 |
| passwd_sha | Passwort neuer Client (SHA-1-verschlüsselter HexCode) | |
| info | Beschreibung des Users (Sachgebiet, Fachgebiet) | SG 22 |

Diese Tabelle kann manuell gepflegt werden und bildet die Grundlage für die Benutzerverwaltung in SuperX. Die Tabelle wird mit mehreren Tabellen verknüpft, z.B. user_group_bez.

Die Benutzerpassworte werden sha-verschlüsselt gespeichert. Sie können auch externe Daten in diese Tabelle füllen, z.B. aus einer anderen Benutzerverwaltung. Die Passwort-Verschlüsselung müssen Sie allerdings selbst vornehmen. Unter PostgresSQL kann dieses Verfahren elegant automatisiert werden, dort gibt es mit dem Paket pgcrypto eine Möglichkeit, SHA-Digests scriptgesteuert zu erzeugen. Die Syntax lautet:

```
select encode(digest('<<Klartext-Passwort>>', 'sha1'), 'hex');
```

#### 4.1.1.2Tabelle user_masken_bez

Rechte einzelner User für einzelne (Abfrage-)Masken werden in der Tabelle `user_masken_bez` gespeichert:

**Tabelle user_masken_bez**

| Feld | Erläuterung | Beispiel |
|---|---|---|
| userinfo_id | Nummer des Benutzers<br>(entspricht dem Feld tid in der Tabelle `userinfo`) | 1 |
| maskeninfo_id | Nummer der Maske<br>(entspricht dem Feld tid in der Tabelle `maskeninfo` s.u.) | 10050 |

Der *testuser*, der den tid-Eintrag 1 in der Tabelle `userinfo` hat, bekommt Zugriffsrechte für die (Abfrage-)Maske mit der Nummer 10050 (Studierende allgemein).

Falls Sie Einzelrechte für (Abfrage-)Masken vergeben wollen, machen Sie einen Eintrag in der Tabelle `user_masken_bez`.

#### 4.1.1.3Tabelle sachgebiete

Damit man nicht jedem Benutzer für jede (Abfrage-)Maske einzeln Rechte geben muss, gibt es sog. Sachgebiete.

In SuperX gibt es u.a. die Sachgebiete

- Studierende/Prüfungen
- Personal/Stellen
- Gebäude/Räume/Flächen
- Kennzahlen
- Haushalt.

Diese Sachgebiete finden sich in der Tabelle `sachgebiete`.

**Tabelle sachgebiete**

| Feld | Erläuterung | Beispiel |
|---|---|---|
| tid | Nummer des Sachgebiets | 1 |
| name | Name des Sachgebiets | Studierende/Prüfungen |

#### 4.1.1.4Tabelle sachgeb_maske_bez

In der Tabelle `sachgeb_maske_bez` wird die Beziehung von (Abfrage-)Masken zu Sachgebieten festgelegt.

**Tabelle sachgeb_maske_bez**

| Feld | Erläuterung | Beispiel |
|---|---|---|
| sachgebiete_id | Nummer des Sachgebiets<br>(entspricht dem Feld tid in der Tabelle sachgebiete) | 1 |
| maskeninfo_id | Nummer der Maske<br>(entspricht dem Feld tid in der Tabelle maskeninfo s.u.) | 10050 |

Das Beispiel bedeutet, dass die Abfrage Studierende Allgemein (Nr. 10050) zum Sachgebiet Nr.1 (Studierende/Prüfungen) gehört.

#### 4.1.1.5Tabelle user_sachgeb_bez

Man kann dann einem Benutzer Zugriffsrechte für ein komplettes Sachgebiet (z.B. Studierende/Prüfungen) geben. Dazu macht man einen Eintrag in die Tabelle `user_sachgeb_bez`.

**Tabelle user_sachgeb_bez**

| Feld | Erläuterung | Beispiel |
|---|---|---|
| userinfo_id | Nummer des Users<br>(entspricht dem Feld tid in der Tabelle userinfo) | 1 |
| sachgebiete_id | Nummer des Sachgebiets<br>(entspricht dem Feld tid in der Tabelle sachgebiete) | 1 |

Das Beispiel bedeutet,dass der User Nr. 1 (*testuser*) alle Masken aufrufen darf, die zum Sachgebiet Nr. 1 (Studierende/Prüfungen) gehören. Da über die Sachgebietseintragungen Rechte verwaltet werden, sollte hier pro Maske nur eine Eintragung erfolgen (im Gegensatz zum alten Client).

#### 4.1.1.6Tabelle user_institution

Weiterhin kann festgelegt werden, für welche Institutionen ein Benutzer Informationen erhalten darf.

Die Tabelle `user_institution` ordnet die Leserechte einer Person auf die Institutionen im Organigramm zu. Wenn ein User alle Institutionen einsehen darf, dann ist es notwendig, einen Datensatz zum User mit dem Wert 0 im Feld `ch110_institut` hinzufügen (Gültigkeitszeitraum beachten!). Der Übersichtlichkeit halber sollte man im Feld Name dann "Alle" eintragen.

In der Downloadversion des Kernmoduls enthält diese Tabelle zwei Beispielsätze: der User "admin" hat die Leserechte auf einen Fachbereich und auf alles.

**Tabelle user_institution**

| Feld | Erläuterung | Beispiel |
|---|---|---|
| userid | Nummer des Benutzers<br>(entspricht dem Feld tid in der Tabelle `userinfo`) | 1 |
| name | Name der Institution<br>(entspricht dem Feld name in der Tabelle `organigramm`)<br>Eingaben nicht unbedingt erforderlich | TestFB |
| ch110_institut | Nummer der Institution<br>(entspricht dem Feld key_apnr in der Tabelle `organigramm`) | 11 |
| hierarchie | Hierarchieebene (0,1,2,3)<br>[wird in Zukunft für COB-Abfragen benutzt] | |
| gueltig_seit | Gültigkeit der Rechte: Anfang | 1.1.2001 |
| gueltig_bis | Gültigkeit der Rechte: Ende | 31.12.2001 |
| lehre | Freigabe einer Lehrheit bei lehrbezogenen Abfragen Wenn ein User z.B. keine Rechte für die Lehreinheit Physik hat, sondern nur für das untergeordnete Institut A, kann man hier beim Eintrag der Lehreinheit Physik das Feld auf 1 setzen, damit der User bei lehrbezogenen Abfragen (z.B. Mittelberechnung) trotzdem die Rechte für die Lehreinheit erhält. In allen anderen Fällen muss hier null stehen. | null |

Der *testuser* erhält im Beispiel das Recht den TestFB (ch110_institut - key_apnr=11) im Zeitraum vom 1.1.2001 bis zum 31.12.2001 einzusehen.

Die Masken, die einen Organigramm-Button enthalten, arbeiten mit den Prozeduren `sp_user_orga` bzw. `sp_user_orga_child`, die die Institutionen zusammensuchen und in eine temporäre Tabelle schreiben.

## 4.1.2Gruppenverwaltung

Man braucht nicht jedem User einzeln Rechte für (Abfrage-)Masken geben, sondern kann dies auch über die Zugehörigkeit zu einer Gruppe erledigen.

### 4.1.2.1Tabelle groupinfo

Die Gruppennamen werden in der Tabelle `groupinfo` festgelegt.

**Tabelle groupinfo**

| Feld | Erläuterung | Beispiel |
|---|---|---|
| tid | Nummer der Gruppe | 0 |
| name | name der Gruppe | Dezernenten |

#### 4.1.2.2 Tabelle user_group_bez

Die Zugehörigkeit eines Users zu einer Gruppe wird in der Tabelle `user_group_bez` festgelegt.

**Tabelle user_group_bez**

| Feld | Erläuterung | Beispiel |
|---|---|---|
| userinfo_id | Nummer des Users<br>(entspricht dem Feld tid in der Tabelle `userinfo`) | 1 |
| groupinfo_id | Nummer der Gruppe<br>(entspricht dem Feld tid in der Tabelle `groupinfo`) | 1 |

Der *testuser* (Nr. 1) wird zur Gruppe 1 (Dezernenten) gerechnet.

Die Gruppenrechte werden in den Datenbank-Tabellen `group_masken_bez` und `group_sachgeb_bez` gespeichert.

#### 4.1.2.3 Tabelle group_masken_bez

In der Tabelle `group_masken_bez` wird festgelegt, auf welche (Abfrage-)Masken die Gruppe Zugriff haben soll.

**Tabelle group_masken_bez**

| Feld | Erläuterung | Beispiel |
|---|---|---|
| groupinfo_id | Nummer der Gruppe<br>(entspricht dem Feld tid in der Tabelle `groupinfo`) | 0 |
| maskeninfo_id | Nummer der Maske<br>(entspricht dem Feld tid in der Tabelle `maskeninfo` s.u.) | 10050 |

Das Beispiel besagt, dass die Gruppe der Dezernenten (und damit alle User, die dieser Gruppe zugeordnet sind), Zugriffsrechte für die (Abfrage-)Maske 10050 (Studierende allgemein) hat.

#### 4.1.2.4 Tabelle group_sachgeb_bez

Wie bei einzelnen Usern kann man auch bei Gruppen Zugriffsrechte auf ganze Sachgebiete festlegen. Dies geschieht in der Tabelle `group_sachgeb_bez`.

**Tabelle group_sachgeb_bez**

| Feld | Erläuterung | Beispiel |
|---|---|---|
| groupinfo_id | Nummer der Gruppe<br>(entspricht dem Feld tid in der Tabelle `groupinfo`) | 0 |
| sachgebiete_id | Nummer des Sachgebiets<br>(entspricht dem Feld tid in der Tabelle `sachgebiete`) | 1 |

Das Beispiel zeigt die Freigabe des Sachgebiets 1 (Studierende/Prüfungen) für die Gruppe der Dezernenten. Die Reihenfolge der Berücksichtugng von Rechten ist wichtig. Werden innerhalb eines Sachgebiets Einzelrechte auf eine Abfrage vergeben, dann müssen bei jeder neuen Abfrage in diesem Sachgebiet wieder Einelrechte vergeben werden.

### 4.1.3Zugriffsprotokollierung

Alle fehlgeschlagenen Anmeldeversuche an die Datenbank SuperX werden protokolliert (proto_fkt_id=2). Die Protokollierung dient nur der Überwachung der Autorisierung des Anmeldenden; darüber hinaus findet keinerlei Aufzeichnung von Benutzeraktivitäten o. ä. statt. Die maximale Anzahl der Anmeldeversuche ist für jeden Benutzer einstellbar (Tabelle ‚userinfo', Feld ‚max_versuch') und ist standardmäßig auf 5 Versuche eingestellt. Wird die maximale Anzahl überschritten, so wird die Benutzerkennung gesperrt. Der SuperX-Administrator könnte sich zusätzlich ein per Cronjob aufzurufendes Skript einrichten, dass bei Häufung von fehlgeschlagenen Anmeldungen ein Warnemail verschickt.

#### 4.1.3.1Die Tabelle protokoll

Die Tabelle Protokoll enthält die Protokollsätze der Zugriffe auf SuperX.

| Attributname | Bedeutung | Typ |
|---|---|---|
| protokoll_id | ID des Protokollsatzes | serial |
| proto_fkt_id | Nummer der Protokollfunktion (siehe Tabelle proto_funktion) | smallint |
| userinfo_id | Benutzer-ID (aus Tabelle userinfo) | integer |
| ip_adresse | IP-Adresse des Benutzers | char(16) |
| client_name | Rechnername des Benutzers bzw. Fehlermeldung, wenn keine Netzverbindung möglich war (WINSOCKET -Fehler etc.) | char (255) |
| zeitpunkt | Zeitpunkt des protokollierten Ereignisses | datetime year to second |

#### 4.1.3.2Die Tabelle proto_funktion

Diese Tabelle enthält Funktionen, die protokolliert werden.

| Feld | Bedeutung | Typ |
|---|---|---|
| proto_fkt_id | Nummer der Protokoll-funktion | smallint |
| proto_fkt | Name der Protokoll-funktion | char(20) |

Inhalt der Tabelle `proto_funktion`:

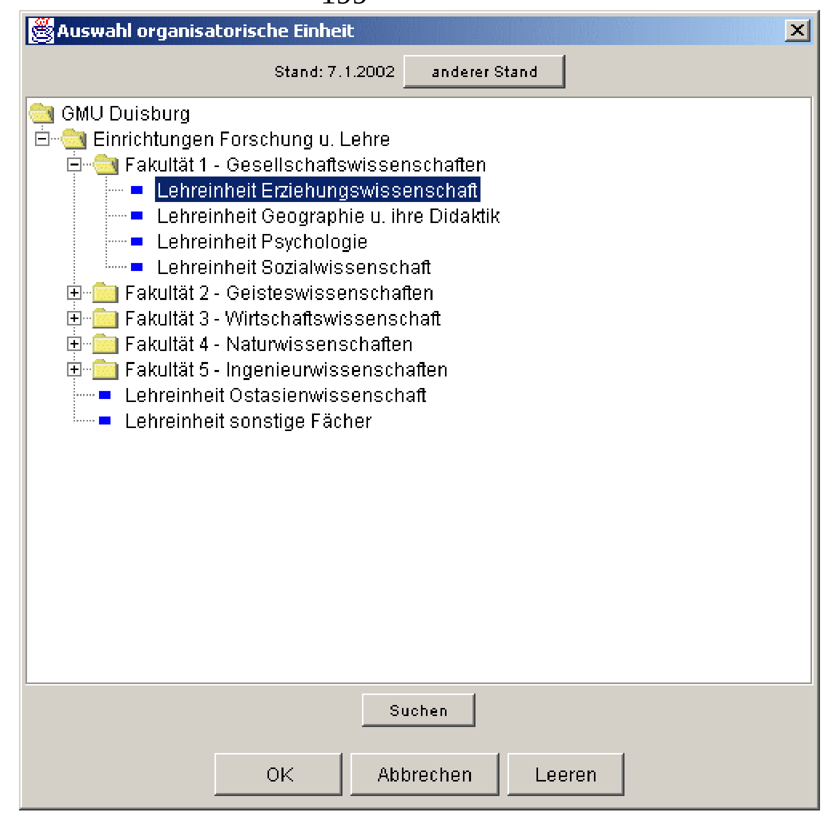
| proto_fkt | Bedeutung |
|---|---|
| LOGIN | Benutzer ist angemeldet |
| LOGIN_FAIL | falsches Paßwort eingegeben |
| LOGIN_LOCKED | Kennung gesperrt (Versuche> max_versuch) |
| LOGOUT | Benutzer hat sich abgemeldet |
| CH_PASSWD_OLD | Paßwort im Paßwortänderungsdialog eingegeben |
| CH_PASSWD_NEW | Neues Paßwort im Paßwortänderungsdialog vergeben |
| CH_PASSWD_FAIL | Falsches Paßwort im Paßwortänderungsdialog eingegeben |

## 4.2Das Organigramm

Das Organigramm stellt eine integrierende Sicht für verschiedene Datenquellen zusammen und ist somit die Voraussetzung für eine integrierte Betrachtung. Alle Einrichtungen, Institutionen und Projekte sind im Organigramm hierarchisch angeordnet.

Die folgende Abbildung zeigt ein Beispiel für ein Organigramm:

Wie im Themenbaum können Sie durch einen hierarchischen Baum navigieren.




Das Organigramm wird vom Java-Applet aus der Tabelle `organigramm` unter Berücksichtigung der Userrechte aufgebaut.

### 4.2.1Die Tabelle Organigramm

Das Kernmodul enthält bei Auslieferung das Organigramm der Universität Duisburg als Beispiel. Die Datensätze können nach der Installation als Vorlage dienen.

| Feld | Erläuterung | Beispiel |
|---|---|---|
| tid | Interne Nummer | |
| key_apnr | Institutionennummer (z.B. im MBS) | "1" |
| parent | key_apnr der übergeordneten Institution | "0" |
| drucktext | Kurzer Text | Einr. Forsch. und Lehre |
| name | Name der Institution | Einrichtungen Forschung und Lehre |
| ebene | Hierarchieebene | 1 |
| lehre | Ist diese Institution relevant für Auswertungen im Bereich Lehre (1=ja, 0=nein); das Feld darf nicht leer sein | 1 |
| gueltig_seit | Beginn des Gültigkeitszeitraums | 1.10.2001 |
| gueltig_bis | Ende des Gültigkeitszeitraums | 31.12.2999 |
| orgstruktur | In diesem Feld wird festgelegt, dass es ich bei einem Eintrag um eine besondere Institution handelt.<br>Wenn es sich um einen Fachbereich handelt, trägt man 20 ein, wenn es sich um eine Lehreinheit handelt 30. | |

Bei der Gültigkeit bestehen einige Abhängigkeiten. Z.B. müssen bei Lehreinheiten als Anfang/Ende jeweils die Semestertermine genommen werden (also Lehreinheit alt gültig_bis 30.9.2001, Lehreinheit neu gültig_seit 1.10.2001).

Generell sollten Einrichtungen, die unbegrenzt gültig sind, im Feld gueltig_bis das Datum "31.12.2999" haben.

Erläuterung des parent-Felds:

Das Parent-Feld gibt die key_apnr der übergeordneten organisatorischen Einheit an. Das root-Element des Baums besitzt als parent einen null-Wert. Das Organigramm darf demenstprechend nur ein Element haben, dass keinen parent besitzt. Hier sollte grundsätzlich die Hochschulnummer eingetragen werden die auch in anderen HIS-Programmen verwendet wird. In Duisburg ist dies z.B. der Basiseintrag key_apnr = "70" (*GMU Duisburg*). Es ist praktisch der oberste "Knoten" im Baum. Dann werden alle Einträge in der Tabelle organigramm gesucht, die parent="70" haben – also direkte Kinder des Basiseintrags.

Hier findet sich u.a. *Einrichtungen Forschung und Lehre* (key_apnr="7", parent="70"). Dieser Eintrag hat wiederum u.a. folgende Kinder:

*Fakultät 1* (key_apnr="1100", parent="7")

*Fakultät 2* (key_apnr="1200", parent="7")

*Faktultät 3* (key_apnr="1300", parent="7")

usw.

Die weiteren Äste des Baums werden rekursiv abgefragt.

### 4.2.2Füllen des Organigramms

Das Organigramm kann von Anwendern, die das "alte" SuperX bereits nutzen, relativ einfach importiert werden. Bei Neuinstallation von SuperX kann man (falls vorhanden) von der Institutionentabelle in MBS ausgehen (im SuperX-MBS-Modul lautet diese Tabelle `mbs_inst`). Der Imort würde lauten

```
alter table organigramm modify (tid serial);
insert into organigramm select 0, inst_nr, uebinst_nr, lname1, lname2, "", 0, key_von, key_bis,
orgstruktur from inst;
alter table organigramm modify (tid integer, key_apnr char(10), parent char(10));
```

Danach könnte man diese recht "flache" Hierarchie nachbearbeiten bzw. nicht gewünschte Unterorganisationen streichen. Nachträglich müssen alle Organisationseinheiten, die oberhalb einer Lehreinheit liegen (z.B. Fakultäten), sowie die Lehreinheiten selbst, auf lehre = 1 stehen.

### 4.2.3Die Prozedur sp_user_orga

Die Prozedur `sp_user_orga` sucht die Institutionen, die ein User sehen darf, und bereitet sie in einem temporären Organigramm auf (siehe Organigramm).

```
sp_user_orga(userid integer default -1, p_datum date default today, p_lehre smallint)
```

Liefert für angegebenen Stand alle org. Einheiten zurück, die ein Benutzer sehen darf,

| | |
|---|---|
| p_lehre=0 | alle org. Einheiten, für die ein Benutzer Rechte hat |
| p_lehre=1 | nur org. Einheiten aus dem Bereich Lehre, für die der Benutzer Rechte hat |
| p_lehre =2 | Benutzer darf alle org. Einheiten im Bereiche Lehre sehen |

### 4.2.4Die Prozedur sp_user_orga_child

Die Prozedur `sp_user_orga_child` generiert die Tabelle `tmp_ch110institut`, die wiederum in der Abfrage aufgerufen wird und die alle Institutionen enthält, die ein User sehen darf und ausgewählt hat, d.h. den aktuellen „Ast“ des Users im Organigramm.

```
sp_user_orga_child(userid integer default -1, p_datum date default today, p_lehre smallint,
p_key_apnr integer, p_erlaubt smallint)
```

Liefert für einen angegebenen Stand alle Untereinheiten einer org. Einheit, die ein Benutzer einsehen darf.

| | |
|---|---|
| p_lehre=0 | alle, |
| p_lehre=1 | nur die für den Bereich Lehre, |

Aufruf steht im select_stmt (Tabelle `maskeninfo`), Änderungen dort

| | |
|---|---|
| p_erlaubt = 0 | Benutzer darf Einheit nicht komplett einsehen, nur ein oder mehrere untergeordnete Einheiten (z.B. nicht der gesamte FB6 – nur Geographie) |
| p_erlaubt = 1 | Benutzer darf die gewählte Einheit mit allen Untereinheiten einsehen.<br>p_erlaubt wird vom Applet gesetzt. |

Beispielaufruf in einem SQL-Script:

```
execute procedure sp_user_orga_child ( <<USERID>>,<<Organigramm-Stand>>, 0, /* <<Institution>>,
<<erlaubt>>)
```

Variablen in << >> werden vom Applet vor der Ausführung z.B. wie folgt ersetzt

User1, Fachbereich 6 (Interne Nummer = 6), Stand 1.5.2002, den der User komplett einsehen darf:

```
execute procedure sp_user_orga_child (1, "1.5.2002", 0, "6", 1)
```

Achtung: Diese Prozedur ist in PostgreSQL bisher noch nicht unter Berücksichtugng der Userrechte in user_institution implementiert.

## 4.3Die SuperX-Auswertungen

Im folgenden werden die grundlegenden Tabellen für die Verwaltung der SuperX-Auswertungen bzw. Abfragen erläutert.

### 4.3.1Die Tabelle Maskeninfo

Basis einer SuperX-Abfrage ist ein Eintrag in der Tabelle `maskeninfo`. Eigene Masken müssen immer in einem definierten Nummernkreis liegen (z.B. >=10.000, <20000) ) und Zehnerzahlen sein (z.B. 10050).

**Tabelle maskeninfo**

| Feld | Erläuterung | Beispiel |
|---|---|---|
| tid | Interne Nummer | 10050 |
| name | Name der (Abfrage-)Maske | Studierende Allgemein |
| select_stmt | SQL-Audrücke, die die Abfrage durchführen | SQL-Statement (s) |
| xil_proplist | beschreibt den Aufbau der Ergebnistabelle (s.u.) | |
| chart_xtitel | für graphische Darstellung der Ergebnisse [derzeit nur im alten Client] | |
| chart_ytitel | für graphische Darstellung der Ergebnisse [derzeit nur im alten Client] | |
| erlaeuterung | Erklärungstext zur Maske | |
| cleanup_stmt | SQL-Ausdruck nach select_stmt, z.B. um temporäre Tabellen wieder zu löschen | drop table tmp_stud; |
| default_file | [wird derzeit nur vom alten Client genutzt] | |
| macro | [wird derzeit nur vom alten Client genutzt] | |
| breite | Breite der Maske in Pixel | |
| hoehe | Höhe der Maske in Pixel | |
| ampel | [wird derzeit nur vom alten Client genutzt] | |
| hilfe | Kennzeichen, ob Java-Hilfetext vorliegt (1=ja, 0=nein) | |
| hinweis | Erläuterungstext zur Ergebnistabelle, wird im Kopf angezeigt | <<SQL>> select erlaeuterung from koepfe_oder_faelle where apnr = "<<Köpfe oder Fälle>>" |

### 4.3.1.1SQL-Scripte

Die für die Suchanfrage einer Maske notwendigen SQL-Ausdrücke sind in der Tabelle `maskeninfo` im Blob-Feld select_stmt abgelegt.

Ein kleines Beispiel soll die Besonderheiten der SuperX-Suchanfragen erläutern.

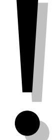
| Feld | Eintrag |
|---|---|
| name | Auslastung |
| select_stmt | select lehreinheit, export, auslastquote from auslastung where jahr = <<Jahr>><br>/*and lehreinheit = <<Lehreinheit>> */<br>into temp tmp_auslastung with no log;<br>select * from tmp_auslastung order by 1 ; |
| cleanup_stmt | drop table tmp_auslastung; |

Es handelt sich hierbei um eine Maske zur Bestimmung der Auslastung einer Lehreinheit. Auf der Auswahlmaske gibt es 2 Felder: Jahr und Lehreinheit. Jahr ist ein obligatorisches, Lehreinheitein fakultatives Eingabefeld.

Für jedes Eingabefeld gibt es im select_stmt eine Variable <<  >>, die beim Auswerten der SQL Anweisungen durch den Inhalt des Feldes ersetzt wird. <<Jahr>> wird durch das vom User gewählte Jahr ersetzt. Handelt es sich wie bei <<Lehreinheit>> um ein fakultatives Eingabefeld, so kann der Feldinhalt

leer sein. In diesem Fall wird zusätzlich der Teil der SQL-Anweisung auskommentiert (/*...*/), in dem die entsprechende Variable vorkommt (zwischen 2 Kommentarklammern (/*...*/) muss genau eine Feldvariable stehen!). Falls eine Lehreinheit vom User ausgewählt wird (z.B. 50000=Psychologie) wird die Zeile `and lehreinheit =50000` mit ausgeführt. Wenn keine Lehreinheit ausgewählt wurde, bleibt sie unberücksichtigt und man erhält einen Gesamtwert über alle Lehreinheiten.

Der Variablen <<UserID>> kommt eine Sonderbedeutung zu: Sie ist im Applet als verborgenes Feld vorhanden. Für <<UserID>> wird die Nummer des Benutzers eingesetzt (vgl. sp_user_orga_child, s.o.).



Wichtig:Die **letzte** SQL-Anweisung muss ein select-Ausdruck sein, der das Ergebnis der Suchanfrage liefert. Das Ergebnis steht in unserem Beispiel in der temporären Tabelle tmp_auslastung. Diese Tabelle muss nach der Ausführung des select-Ausdrucks noch entfernt werden. Dafur gibt es das Feld cleanup_stmt, dessen Inhalt nach Ausführung von select_stmt ausgewertet wird.

#### 4.3.1.2Aufbau der Ergebnistabelle

Das Suchergebnis wird in einer Ergebnistabelle auf einer speziellen Suchergebnismaske dargestellt. Die Definition der Ergebnistabelle geschieht durch besondere Tags, die im Feld `xil_proplist` gespeichert werden.

Wichtig ist, dass für die Ergebnistabelle die Anzahl der selektierten Felder größer sein darf als die Anzahl der COLUMNS in XIL-List sein, aber nicht umgekehrt. Am einfachsten ist es, die Tabellendefinition einer bestehenden Maske zu kopieren und dann anzupassen.

Ein Beispiel für die Abfrage Aufnahmekapazität im aktuellen Studienjahr:

```
^XIL List\
   sizable_columns horizontal_scrolling\
   white_space_color=COLOR_WHITE\
   drop_and_delete movable_columns fixed_columns=1\
   min_heading_height=50\
Column CID=0 heading_text="Lehreinheit / Studiengang" center_heading\
   row_selectable heading_platform readonly\
   width=35 text_size=50\
Column CID=1 heading_text="Aufnahme-\\n kap. o. Ber.\\n Schwundquote" center_heading\
   row_selectable col_selectable rightJust heading_platform readonly\
   width=14\
Column CID=2 heading_text="Aufnahme-\\n kap. m. Ber.\\n Schwundquote" center_heading\
   row_selectable col_selectable rightJust heading_platform readonly\
   width=14\
Column CID=3 heading_text="Studierende\\n im 1. FS\\n im Studienj.  " center_heading\
   row_selectable col_selectable rightJust heading_platform readonly\
   width=11\
Column CID=4 heading_text="1. FS / \\n Aufn.kap o.\\n Schwund in %" center_heading\
   row_selectable col_selectable rightJust heading_platform readonly\
   width=14\
Column CID=5 heading_text="1. FS / \\n Aufn.kap m.\\n Schwund in %" center_heading\
   row_selectable col_selectable rightJust heading_platform readonly\
   width=14\
Column CID=6 heading_text="Studier.\\n in RSZ\\n im WS" center_heading\
   row_selectable col_selectable rightJust heading_platform readonly\
   width=8\
Column CID=7 heading_text="Studier.\\n gesamt\\n im WS" center_heading\
   row_selectable col_selectable rightJust heading_platform readonly\
   width=8 \
@@@
```

Die Attribute zu den Felder werden wie folgt interpretiert:

| Attribut | Erläuterung |
|---|---|
| heading_text | Spaltenüberschrift |
| center_heading | Zeilenausrichtung der Überschrift zentriert |
| row_selectable | Zeile ist selektierbar (z.B. für Kopie in Zwischenablage) |
| col_selectable | Spalte ist selektierbar |
| rightjust | Zeilenausrichtung der Zellen rechtsbündig |
| heading_platform | wird z.Zt. nicht genutzt |
| readonly | Nicht editierbar |
| height | Höhe der Zelle in Pixel (default ist 12) |
| width | Breite der Zelle in Pixel (default ist Auto) |

### 4.3.1.3Verbindung zur Tabelle felderinfo

Die einzelnen Felder auf einer Maske (z.B. Semester, Lehreinheit, Haushaltsjahr, Köpfe oder Fälle) sind in der Tabelle `felderinfo` abgelegt. Gibt es zu einer Maske der Nummer <TID> n Felder, so haben diese in der Tabelle `felderinfo` die Nummern:

<TID>, <TID>+1, ..., <TID>+n-1

So gehören z.B. zur Maske 10050 "Studierende allgemein", die folgenden Einträge in der Tabelle felderinfo

| tid | name |
|---|---|
| 10050 | Köpfe oder Fälle? |
| 10051 | Semester |
| 10052 | Organisatorische Einheit |
| 10053 | Abschluss |
| 10054 | bis Fachsemester |
| 10055 | Hörerstatus |
| 10056 | Hochschulzugangsberechtigung |
| 10057 | Staatsangehörigkeit |
| 10058 | Aggregierung? |

Gibt es mehr als 10 Felder auf einer Maske, so entfällt die entsprechende Zehnerzahl als Nummer für eine andere Maske, man sollte aber sicherheitshalber eine entsprechenden Eintrag in  machen.

### 4.3.2Tabelle Felderinfo

In der Tabelle `felderinfo` sind die einzelnen Auswahlfelder einer Maske abgelegt (s.o.).

Die Lage der Felder auf der Maske wird durch die Attribute x und y bestimmt. Der Ursprung des Koordinatensystems eines Fensters ist die obere linke Ecke, die y-Achse geht nach unten. Die Felder werden mit dem Offset (z.B. 60,60) platziert.

**Tabelle felderinfo**

| Feld | Erläuterung | Beispiel |
|---|---|---|
| tid | Interne Nummer | |
| name | Name des Feldes auf der Maske | |
| nummer | Nummer des Feldes auf der Maske | |
| x | x-Position auf der Maske | |
| y | y-Position auf der Maske<br>**y = 0**: Gehe in die nächste Zeile<br>**y = -1**: Bleibe in der aktuellen Zeile<br>**sonst**: Nehme y als absoluten (auf die Maske bezogenen) y-Wert | |
| buttonbreite | Die Feldnamen werden durch SuperX rechtsbündig angeordnet. Die dafür vorgesehene Breite wird mit dem Attribut buttonbreite definiert und in Pixel angegeben. Der Wert 100 reicht in den meisten Fällen aus.<br>buttonbreite = 0 : Übernehme die entsprechenden Werte vom vorhergehenden Feld | |
| feldbreite | Die Breite des Eingabefeldes wird mit feldbreite bestimmt. Häufige Wer1e sind 70 für kurze Felder (z.B. Rechnungsjahr) oder 200 für längere Felder (z.B. Institution).<br>feldbreite = 0 : Übernehme die entsprechenden Werte vom vorhergehenden Feld | |
| zeilenanzahl | Für die Anzahl der Zeilen des Eingabefeldes gibt es das Attribut zeilenanzahl.<br>**zeilenanzahl = 1**: Es kann höchtens ein Wert im Eingabefeld angegeben werden.<br>**zeilenanzahl> 1**: Es können mehrere Werte im Eingabefeld angegeben werden. Eingabefeld besitzt Scroll-Leiste. Damit das Eingabefeld nicht zu groß wird, reicht zeilenanzahl = 3 aus.<br>**Achtung**: Bei Mehrfachauswahlfeldern, die als Werte integer-Schlüssel zurückgeben sollen, muss beim folgenden Attribut typ der Wert sql eingetragen werden. In SuperX werden die einzelnen Werte mit "," getrennt, z.B.<br>select * from groupinfo where tid in <<Gruppe>><br>wird aufgelöst nach:<br>select * from groupinfo where tid in (1,4,6);<br>Wenn der Typ nicht sql, sondern Character ist, geben Sie als Typ char ein; der Select wird dann aufgelöst nach:<br>select * from groupinfo where tid in ('1','4','6'); | |

| typ | In SQL-Ausdrucken wird zwischen numerischen und alphanumerischen Werten unterschieden. Alphanumerische Werte müssen mit Hochkommas versehen werden. Aus diesem Grund gibt es in SuperX verschiedene Typen von Feldern:<br>**typ = integer**: Es werden für dieses Feld in dem SQL-Ausdruck der Suchanfrage keine Hochkommas eingesetzt.<br>**typ = char**: Im `select_stmt` werden automatisch Hochkommata um den Variablenwert gesetzt. Bei Mehrfachauswahlfeldern gibt es eine Sonderbehandlung (s.o.).<br>**typ = sql**: Das Ergebnis des Feldes liefert einen SQL-Ausdruck. Im `select_stmt` werden keine Hochkommas eingefügt.<br>**typ = date**:Es werden für dieses Feld in dem SQL-Ausdruck der Suchanfrage date('...') eingesetzt, bei PostgreSQL lautet die Funktion date_val('...').<br>**typ = decimal**:Der eingegebene Wert (mit "," als Dezimaltrennzeichen) wird im SQL-Ausdruck der Suchanfrage mit "." als Dezimaltrennzeichen versehen. | |
|---|---|---|
| laenge | zur Zeit nicht genutzt | |
| obligatorisch | Eingabe kann zwingend (obligatorisch =1) oder freiwillig sein (obligatorisch = 0) | |

| art | In den operativen Systemen wird oft mit Schlüsselwerten (z.B. Institutsnummern) gearbeitet. Um selbsterklärend zu sein, werden in SuperX nur intern diese Nummern verwendet. Nach außen sieht der Anwender den Klartext (z.B. den Institutsnamen). Für die Felder muss unterschieden werden, ob eine Nummemausprägung existiert. Daher gibt es verschiedene Arten von Feldern:<br>**art = 0**: SuperX verwaltet nur den Wert des Eingabefeldes. In diese Felder kann man immer direkt Werte eingeben. Die Eingabe über einen Dialog ist wahlweise möglich (siehe Abschnitt Dialogsteuerung).<br>**art = 1**: Eine Dialogbox wird geöffnet, die eine Liste mit Auswahlmöglichkeiten anzeigt. Die Datenquelle ist eine Tabelle oder ein SQL-Ausdruck, wobei die erste Spalte unsichtbar ist und den Rückgabewert der Dialogbox liefert. In diese Felder kann man nicht direkt, sondern nur über einen Dialog Werte eingeben.<br>**art = 2**: Genau wie art = 1. Die Auswahlwerte des Dialogs können jedoch nur durch Angabe einer Stored Procedure bestimmt werden.<br>**art = 3**: Feld zur Auswahl einer Datei mit Hilfe des plattformspezifischen Dateiauswahldialogs. Eine direkte Eingabe ist nicht möglich, das Feld ist nur einzeilig. Wird in Version 2.0 des Applets noch nicht umgesetzt.<br>**art = 4**: Es handelt sich um ein Feld zur Auswahl einer Institution oder Person. Dazu wird ein spezieller Dialog geöffnet, der die hierarchische Struktur der Hochschule widerspiegelt. Dabei werden alle Institutionen angezeigt, für die der Benutzer Zugriffsrechte hat.<br>**art = 5**: SuperX verwaltet nur den Wert des Eingabefeldes. Im Unterschied zu art = 0 ist eine direkte Eingabe nicht möglich. Die Art kann für die Gestaltung von Kommentarzeilen genutzt werden.<br>**art = 6**: wie art=4, aber es erscheinen nur alle Institutionen im Bereich Lehre, für die der Benutzer Rechte hat (lehre=1 in der Tabelle `organigramm`)<br>**art = 7**: wie art=4, aber es erscheinen alle Institutionen im Bereich Lehre (lehre=1 in der Tabelle organigramm) ohne Rechteeinschränkung<br>**art = 8**: Das Feld dient nur als Label, es erscheint kein Eingabefeld<br>**art = 12**: Sicht, in Spalte relation muss ein SQL stehen, der die tids der gewünschten Sichten aus der Sichtentabelle liefert | |
|---|---|---|
| relation | wird für Dialogsteuerung benötig, s.u. | |
| attribut | Feld in der DB-Tabelle, die man in relation angegeben hat (s. Dialogsteuerung) | |
| default_wert | Vorgabewerte für den Feldinhalt (s.u.) | |

#### 4.3.2.1Dialogsteuerung

Überwiegend erfolgt die Eingabe in die Felder dialoggesteuert, das heißt der Anwender kann aus einem Dialog mögliche Eingabewerte auswählen. In der Tabelle `felderinfo` kann zu jedem Feld definiert wer-

den, wie die möglichen Auswahlwerte des Dialoges lauten. Um selbsterklärend zu sein, sollte der Anwender nur Klartext sehen. Die entsprechende Nummernausprägung verwendet SuperX nur intern. Die Bestimmung der Auswahlwerte eines Dialogs geschieht entweder durch Angabe einer DB-Tabelle und eines dazugehörigen Attributs, durch Angabe einer Stored Procedure oder eines SQL-Ausdrucks. Soll die Feldeingabe mit Hilfe eines Dialoges **nicht** möglich sein, so müssen die Einträge für relation und attribut leer bleiben.

#### 4.3.2.1.1Angabe einer DB- Tabelle

Soll z.B. auf einer Studierenden-Maske das Semester ausgewählt werden können, so steht in der Tabelle `felderinfo`:

| Feld | Eintrag |
|---|---|
| name | Semester |
| relation | semester |
| attribut | eintrag |

In der SuperX-Datenbank gibt es dazu die Tabelle `semester`.

| Feld | Erläuterung | Beispiel |
|---|---|---|
| tid | Interne Nummernausprägung des Semesters | 20011 |
| eintrag | Semester als Klartext | SS 2001 |
| sem_beginn | Datum des Semesterbeginns | 01.04.2001 |
| sem_ende | Datum des Semesterendes | 30.09.2001 |

#### 4.3.2.1.2Angabe einer Stored Procedure

Statt einer DB- Tabelle kann auch eine Stored Procedure angegeben werden. Diese wird beim Maskenaufbau ausgeführt und liefert als Rückgabewerte die Auswahlwerte des Dialogs.

Damit SuperX zwischen einer Tabelle und einer Stored Procedure unterscheiden kann, muss der Name der Stored Procedure mit "sp_" beginnen. Bei Feldern mit art = 2 ist diese Konvention nicht erforderlich.

Häufiges Beispiel sind Felder, deren Eingaben aus der CIF (Central Information File) kommen. Hier die Tabelle `felderinfo` für ein Feld, mit welchem die Dienstart ausgewählt werden soll (in der CIF: hochschulallgemeiner Schlüssel der Nummer 107):

| Feld | Eintrag |
|---|---|
| name | Dienstart |
| relation | sp_cif(0,107) |
| attribut | |

#### 4.3.2.1.3Angabe eines SQL-Ausdrucks

Die Ergebnisse des angegebenen SQL-Ausdrucks sind die Auswahlwerte des Dialogs. Genau wie bei der Definition von Vorgabewerten für Felder muß der SQL-Ausdruck mit "<<SQL>>" beginnen. Beispiel:

| **Feld** | **Eintrag** |
|---|---|
| name | Etage |
| relation | <<SQL>> select distinct geschossnr, druck from baupc-geschoss order by 1; |
| attribut | |

#### 4.3.2.1.4Hinweis für Dialogart 1 und 2

Für Felder bei denen eine Nummernausprägung intern verwendet wird (art = 1, 2) ist zu beachten, dass die Stored Procedure bzw. der SQL-Ausdruck 2 Werte (Nummernausprägung und Klartext) zurückliefern muss. Bei Angabe einer Tabelle müssen entweder 2 Attribute angegeben werden (attribut = A1,A2) oder man gibt nur das Attribut für den Klartext an. In diesem Fall geht SuperX davon, dass das Attribut für die Nummernausprägung "tid" heißt. Die Nummernausprägung muss zuerst angegeben werden. Man kann natürlich auch zugleich Nummernausprägung und Klartext in dem Dialog darstellen:

<<SQL>> select geschossnr, druck || "(" || geschossnr || ")" from baupc-geschoss;

### 4.3.2.2Vorgabewerte für die Felder

Beim Öffnen einer Maske können dem Anwender Vorgabewerte angeboten werden, Dabei handelt es sich entweder um konstante Werte oder um Ergebnisse eines SQL-Ausdrucks:

#### 4.3.2.2.1Konstanten

Ein Feld für die Eingabe eines Rechnungsjahres soll den fest vorgegebenen Wert "2002" besitzen. In der Tabelle `felderinfo` steht:

| **Feld** | **Eintrag** |
|---|---|
| name | Rechnungsjahr |
| default | 2002 |

#### 4.3.2.2.2SQL-Ausdrücke

Viel flexibler ist die Definition des Vorgabewertes mit Hilfe eines SQL-Ausdrucks. Damit kann sowohl auf Werte aus der Datenbank als auch auf das aktuelle Datum zugegriffen werden.

Damit SuperX zwischen Konstanten und SQL-Ausdrücken unterscheiden kann, beginnen letztere mit "<<SQL>>" (Leerzeichen nicht vergessen !). Im folgenden Beispiel lautet der
SQL-Ausdruck für das Vorjahr:

| Feld | Eintrag |
|---|---|
| name | Rechnungsjahr |
| default | `<<SQL>> select (year(today) -1) \|\| "" from xdummy;` |

Die Tabelle xdummy ist eine Tabelle mit einem Satz. Sie dient lediglich dazu, den o.g. SQL-Ausdruck syntaktisch korrekt zu machen.

Für Felder bei denen eine Nummernausprägung intern verwendet wird (art = 1, 2) ist zu beachten, dass der SQL-Ausdruck 2 Werte (Nummernausprägung und Klartext) zurückliefern muss.

Achtung: Vorgabewerte können nicht für Institutions-Felder (art = 4,6,7) angegeben werden. Für mehrzeilige Felder (zeilenanzahl> 1) können lediglich SQL-Ausdrücke angegeben werden.

### 4.3.3Tabelle systeminfo

Die Tabelle `systeminfo` enthält für einzelne Sachgebiete/System, das Datum des letzten Datenupdates.

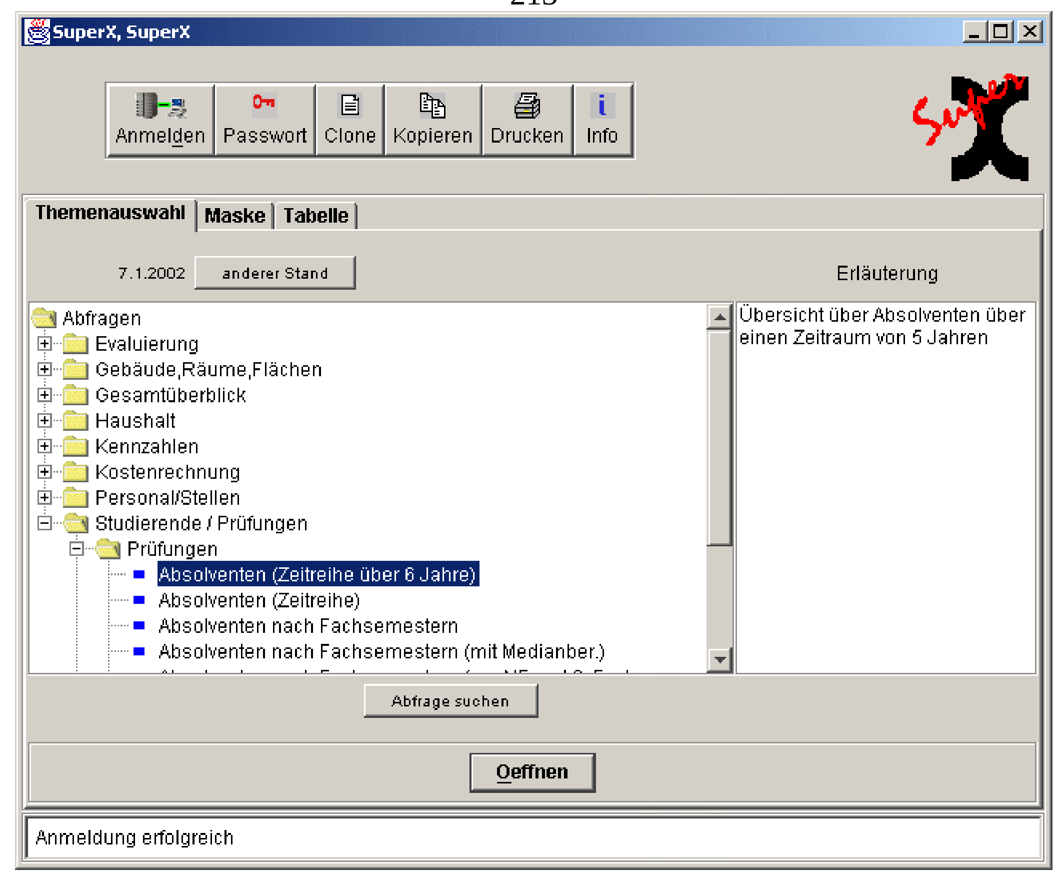
**Tabelle systeminfo**

| Feld | Erläuterung | Beispiel |
|---|---|---|
| tid | interne Nummer | 6 |
| name | Name des Systemteils | Personal/Stellen |
| datum | Datum des letzten Datenupdates | 14.1.2002 |

Dier Eintrag aus der Tabelle `systeminfo` wird über die Tabelle `maske_system_bez` mit der Tabelle `maskeninfo` verknüpft; so lassen sich die Abfragen den Systemen zuordnen.

### 4.3.4Die Tabelle themenbaum

Nach der Anmeldung erhält der User eine Reihe von Auswertungen zur Auswahl in Form eines sog. "Themenbaums". Der Themenbaum wird dynamisch generiert aus der Tabelle `themenbaum`, die alle Auswertungen und deren hierarchischen Zusammenhang enthält, und den spezifischen Rechten, die der Benutzer hat. Die folgende Abbildung zeigt einen Ausschnitt aus einem Beispiel-Themenbaum.




Das Java-Applet erzeugt aus der Tabelle `themenbaum` unter Berücksichtigung der Userrechte die graphische Oberfläche.

Hier werden die Themen und Sachgebiete gesammelt und strukturiert. So kann aus der relativ einfachen Zuordnung von Themen (bzw. Masken) und Sachgebieten eine relativ komplexe Hierarchie gebildet werden.

**Tabelle Themenbaum**

| **Feld** | **Erläuterung** | **Beispiel** |
|---|---|---|
| tid | Interne Nummer | 2 |
| name | Name der Maske bzw. des Sachgebiets | Absolventen |
| maskeninfo_id | ID der Maske<br>(entspricht dem Feld tid in der Tabelle `maskeninfo`)<br>Bei Sachgebieten bleibt dieses Feld leer. | 10140 |
| parent | ID der übergeordneten Maske bzw. des Sachgebiets | 1 |
| sort | Sortiernummer | 10 |
| gueltig_seit | Beginn des Gültigkeitszeitraums | 1.1.2001 |
| gueltig_bis | Ende des Gültigkeitszeitraums | 1.10.3000 |
| erlaeuterung | [Wird nicht benutzt] | |

Der hierarchische Aufbau der Tabelle über das Feld parent entspricht dem der Tabelle `organigramm`. Die folgende Tabelle zeigt ein paar Beispieleinträge.

| tid | name | maskeninfo_id | parent | sort | gueltig_seit | gueltig_bis |
|---|---|---|---|---|---|---|
| 5 | Kennzahlen | | 1 | | 01.01.1900 | 01.01.3000 |
| 8 | Studierende | | 2 | | 01.01.1900 | 01.01.3000 |
| 6 | Gesamtüberblick | | 5 | | 01.01.1900 | 01.01.3000 |
| 4 | Gebäude,Räume,Flächen | | 1 | | 01.01.1900 | 01.01.3000 |
| 3 | Personal/Stellen | | 1 | | 01.01.1900 | 01.01.3000 |
| 2 | Studierende / Prüfungen | | 1 | | 01.01.1900 | 01.01.3000 |
| 1 | Abfragen | | | | 01.01.1900 | 01.01.3000 |
| 88 | Kostenrechnung | | 1 | | 01.01.1900 | 01.01.3000 |
| 81 | Zeitreihen | | 2 | | 01.01.1900 | 01.01.3000 |
| 91 | Evaluierung | | 1 | | 01.01.1900 | 01.01.3000 |
| 92 | Prüfungen | | 2 | | 01.01.1900 | 01.01.3000 |
| 7 | Haushalt | | 1 | | 01.01.1900 | 01.01.3000 |
| 47 | Flächenarten für Institutionen | 10010 | 4 | | 01.01.1900 | 01.01.3000 |
| 9 | Studierende nach Hörerstatus | 10040 | 8 | | 01.01.1900 | 01.01.3000 |
| 10 | Studierende allgemein | 10050 | 8 | | 01.01.1900 | 01.01.3000 |

Die Maske *Studierende nach Hörerstatus* hat als übergeordneten Knoten das Thema *Studierende*, und dies wiederum das Thema *Studierende/Prüfungen*.

### 4.3.5 Verkettung von Masken: Die Tabelle macro_masken_bez

Im SuperX können einzelne Masken zu einem Bericht kombiniert werden. Dazu wird ein Makro definiert, und die einzelnen Auswertungen werden dem Makro zugeordnet.

| Feld | Erläuterung | Typ |
|---|---|---|
| maskeninfo_id1 | Makro-Auswertung | integer |
| maskeninfo_id2 | Dem Makro zugeordnete Auswertungen | integer |
| nummer | Ordnungsnummer der Zuordnung | |

Die Makrofunktionalität ist in der Version 2.02 vom SuperX-Kernmodul wie folgt umgesetzt:

1. Sie erzeugen eine Maske mit allen relevanten Feldern. Diese Maske benötigt kein `select_stmt` und keine `xil_proplist`, sie dient gewissermaßen als "Hülle" für die eigentlichen Abfragen.
2. Dann ordnen Sie die vorhandenen Abfragen diese Maske in der obigen Tabelle zu. Mit dem Feld `nummer` legen Sie die Reihenfolge fest. Achten Sie darauf, daß die Feldnamen in der Makro-Maske mit denen in den Unter-Masken identisch sind.
3. Der Makromechanismus läuft im XML-Frontend automatisch ab. Die Stylesheet-Einstellungen der Makro-Maske überlagern die der etwa vorhandenen Einzel-Masken.

Ein Beispiel im COB-Modul ist das Makro zum Berichtsblatt MSWF NRW.

## 4.4Einzelne Schlüsseltabellen

Für die Abfragen aus den verschiedenen Basissystemen gibt es ein paar regelmäßig wiederkehrende Anfragen, z.B. Aggregierung nach Quartalen und Halbjahren. Deshalb sind diese im Kernmodul angelegt.

### 4.4.1Die Tabelle schluessel

Die Tabelle schluessel enthält ausschließlich Metadaten zum Betrieb von SuperX; sie enthält schluessel und Erläuterungstexte zu den einzelnen Funktionalitäten in SuperX, z.B. zu den Feldtypen auf einer Maske etc.

Die Tabelle besitzt folgende Struktur

| Feld | Erläuterung | Typ |
|---|---|---|
| id | Tupelidentifier | integer |
| variable | Art der Variable / des Schlüssels | char(50) |
| wert | Codierung oder SQL-Ausdruck | char(255) |
| beschreibung | Kurzer Erläuterungstext | char(255) |
| typ | Variablentyp / Schlüsseltyp | char(255) |
| erlaeuterung | Langer Erläuterungstext | char(255) |

Die folgende Tabelle zeigt die Metadaten zum Aufbau von Feldern einer Maske. Die "Werte" sind wiederum die Schlüssel, die in der Tabelle Felderinfo als Attibute gefüllt werden.

schluessel : Tabelle

| id | variable | wert | beschreibung | typ | erlaeuterung |
|---|---|---|---|---|---|
| 1 | Feldart | 0 | 0-nur Text, direl | | |
| 2 | Feldart | 1 | 1-Nummer+Tex | | Eine Dialogbox wird geöffnet, die eine Liste mit Au |
| 3 | Feldart | 2 | 2-wie 1, aber m | | |
| 4 | Feldart | 3 | 3-Dateiauswahl | | Feld zur Auswahl einer Datei mit Hilfe des plattfor |
| 5 | Feldart | 4 | 4-Institutionsdia | | Es handelt sich um ein Feld zur Auswahl einer Ins |
| 6 | Feldart | 5 | 5-nur Text, nur | | |
| 7 | Feldart | 6 | 6-Institutionsdia | | wie art=4, aber es erscheinen nur alle Institutioner |
| 8 | Feldart | 7 | 7-Institutionsdia | | wie art=4, aber es erscheinen alle Institutionen im |
| 9 | Feldart | 8 | 8-Erläuterungsf | | |
| 10 | Feldtyp | integer | integer | | |
| 11 | Feldtyp | char | char | | |
| 12 | Feldtyp | sql | sql | | |
| 14 | obligatorisch | 0 | 0-Nein | | |
| 15 | obligatorisch | 1 | 1-Ja | | |
| * | | | | | |

Die Schlüsseltabelle aggregierung

Die Tabelle aggregierung wird in Abfragen verwendet, um nach bestimmten Markmalen zu zu summieren oder zusätzliche Kriterien einzufügen.

Die Abfrage Nutzungsprotokolle (intern) benutzt die Tabelle z.B., um auf Zeiträume (Halbjahre, Quartale) einzuschränken.

Die Tabelle besitzt folgende Struktur

| Feld | Erläuterung | Typ |
|---|---|---|
| tid | Tupelidentifier | integer |
| ord | Sortiernummer | smallint |
| name | Beschreibung | char(30) |
| kategorie | Kategorie | char(30) |
| wert | numerischer Wert oder sql-Ausdruck | char(255) |

### 4.4.2Die Schlüsseltabellen cif und cifx

Die Schlüsseltabelle `cif` ist Bestandteil des Kernmoduls und enthält Schlüssel, die in verschiedenen operativen Systemen verwendet werden. Die Tabelle `cifx` ist eine analoge Schlüsseltabelle, die auch alphanumerische Ausprägungen enthält.

Die Tabellen sind das "Herzstück" des Data Warehouse, und möglichst alle Schlüssel sollen darin enthalten sein. Die Art des Schlüssels wird dirch den Wert "key" bestimmt, und prinzipiell ist es möglich, hochschulspezifische und allgemeine Schlüssel zu pflegen. So gibt es in der `cifx` z.B. den Schlüssel:

| | |
|---|---|
| **Schlüsselname** | ch35_ang_abschluss |
| **Key** | 35 |
| **Bedeutung** | Hochschulspezifischer Schlüssel für die angestrebte Abschlussprüfung aus SOS |

Der jeweilige Wert für des Schlüssels steht im feld apnr, und die Kurz- und Langbeschreibungen stehen in den Felder kurz, druck, lang_1 usw.

| Feld | Erläuterung | Typ |
|---|---|---|
| tid | Interne Nummer | serial |
| hs | Hochschul-Nr. (0=Hochschulübergreifend) | integer |
| key | Schlüsselgruppe | smallint |
| apnr | Schlüssel | integer |
| d_akt_von | Datum von | date |
| d_akt_bis | Datum bis | date |
| kurz | Kurzbeschreibung | char(10) |
| druck | Drucktext | varchar(30) |
| lang_1 | Langbeschreibung 1 | char(50) |
| lang_2 | Langbeschreibung 2 | char(50) |
| lang_3 | Langbeschreibung 3 | char(50) |

Folgende Schlüsselgruppen sind z.B. in SuperX enthalten (Schlüssel in der `cifx` sind gesondert gekennzeichnet):

| key | hs | Bedeutung | Schlüsseltabelle | Herkunft System | Herkunft Tabelle |
|---|---|---|---|---|---|
| 12 | 0 | Staat | | | |
| 13 | <>0 | Familienstand | cifx | | |
| 27 | <>0 | Grund Beurlaubung | | | |
| 30 | 0 | Studienfach | cifx | | |
| 30 | <>0 | Studienfach | cifx | | |
| 35 | 0 | HS-Abschluss | cifx | | |
| 35 | <>0 | HS-Abschluss | cifx | | |
| 36 | 0 | Hochschule | cifx | SVA / COB | k_hochschule |
| 39 | <>0 | Vertiefungsrichtung | cifx | | |
| 40 | <>0 | Studientyp | | | |
| 62 | <>0 | Grund Exmatrikulation | | | |
| 86 | 0 | Dienstverhaeltnis | | | |
| 90 | <>0 | Fakultaet fuer Wahlen | | | |
| 95 | 0 | Anrede / Titel | cifx | | |
| 106 | 0 | Beurlaubungsgrund | cifx | | |
| *107 | 0 | Dienstart | | | |
| 108 | 0 | Amt-/Dienstbezeichnung | cifx | | |
| *109 | 0 | BVL-Gruppe | | | |
| 110 | <>0 | Besch.stelle | | | |
| *115 | 0 | Haushaltsvermerk | | | |
| *116 | 0 | Stellenart | | | |
| 120 | 0 | Bewährungs-, Zeitaufstieg | | | |
| 212 | <>0 | Geldgeber | cifx | | |
| 258 | 0 | Stellung in der HS | | | |
| *259 | 0 | Stellenkategorie | | | |
| 260 | 0 | Grund Ausscheidung/Befristung | cifx | | |
| 261 | 0 | Grund fuer das Besetzungsende | cifx | | |
| 268 | 0 | Staatspruefung-Abschluss | | | |
| *270 | 0 | Besetzungsabweichung | | | |
| 284 | <>0 | Kapitel | cifx | | |
| 286 | 0 | Arbeitszeit | | | |
| *291 | 0 | Personalkategorie | | | |
| 305 | 0 | Sperrkennzeichen | | | |
| 500 | 0 | Mittelschoepfung | cifx | | |
| 501 | 0 | Staat | cifx | | |

Die vorgegebenen Schluessel sind hier mit * gekennzeichnet und dürfen nicht in die `cif` geladen werden. Die Tabelle wird im Zuge der Aufnahme von weiteren Modulen (z.B. SOS und SVA) weiter gefüllt.

### 4.4.3Die Schlüsseltabelle trans_inst

Die organisatorischen Einheiten des Organigramms werden in SuperX gebündelt; mitunter stammen aus den zugrundeliegenden Basissystemen sehr viel detailliertere bzw. "tiefere" Institutionen, die in SuperX nicht unbedingt von Interesse sind, z.B. die Kostenstellen aus HISCOB. Diese Einrichtungen werden in der Tabelle `trans_inst` auf übergeordnete Insitutionen des Organigramms projiziert. Die Tabelle hat folgende Struktur:

| Feld | Erläuterung | Typ |
|---|---|---|
| tid | Tupelidentifier | serial |
| inst_nr | Institutionen- bzw. Kostenstellen-Nummer | char(10) |
| ch110_institut | Übergeordnete bzw. zugeordnete Institution im SuperX-Organigramm | char(10) |
| name | Name der Institution | char(200) |
| gueltig_von | Gültigkeit der Projektion: Datum von | date |
| gueltig_bis | Gültigkeit der Projektion: Datum bis | date |

Bei der Übernahme von Daten aus einem Basissystem wie COB wird dann der Datentabelle das Feld der Institution im SuperX-Organigramm hinzugefügt, in dem die Projektion abgebildet wird: Zum Beispiel die Kostenstelle 1200144 (Werkstatt) wird für den Zeitraum vom 1.1.2001 bis 1.4.2002 der SuperX-Institution 12001 (Lehreinheit Psychologie) zugeordnet:

| inst_nr | 1200144 |
|---|---|
| ch110_institut | 12001 |
| name | Werkstatt |
| gueltig_von | 1.1.2001 |
| gueltig_bis | 1.4.2002 |

Im Ladescript eines Basissystems wird diese Zuordnung dann übertragen:

**Auszug aus dem Ladescript für HISCOB**

```
update cob_busa
set ch110_institut = (select ch110_institut from trans_inst M
where M.inst_nr = cob_busa.instnr and
M.d_gueltig_von <= date ("01." || cob_busa.monat || "." ||
cob_busa.jahr) and
M.d_gueltig_bis >= date ("01." || cob_busa.monat || "." ||
cob_busa.jahr))
where instnr is not null;
```

Neben der eigentlichen Kostenstelle "Werkstatt" finden wir also bei obigem Beispiel im Feld `ch110_institut` die SuperX-Institution "Lehreinheit Psychologie". In allen Statistiken zur Lehreinheit, die auf `cob_busa` beruhen, wird also die Werkstatt stillschweigend hinzugezählt.

### 4.4.4Weitere Schlüsseltabellen

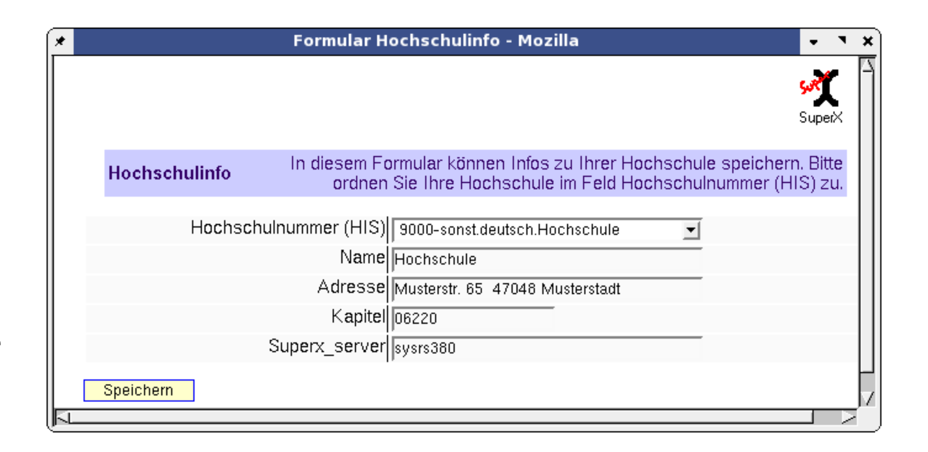
#### 4.4.4.1Tabelle hochschulinfo

Die Tabelle hochschulinfo enthält die Nummer und den Namen der eigenen Hochschule.

Der Schlüssel der Hochschule wird in der Tabelle cif bzw. cifx benutzt, um hochschuleigene Schlüssel von allgmeinen Schlüsseln abzugrenzen.

Sie können die Hochschulinfo in einem DBFORM pflegen; gehen Sie dazu im XML-Frontend auf "Tabelle suchen"-> hochschulinfo. Sie erhalten ein DBFORM mit einem Datensatz:

Wählen Sie Ihre Hochschule aus. Wenn Ihre Hochschule in dem Klappmenü nicht enthalten ist, erfragen Sie die Hochschulnummer bei HIS und tragen sie sie manuell mit einem SQL-Tool in die Tabelle ein.




## 5Hinweise für Entwickler/innen

SuperX enthält verschiedene Formen von Scripten: Das Laden und die Übernahme der Basisdaten sowie die Erzeugung der Hilfstabellen wird von Shell-Scripten erledigt, wie in der Installationsanleitung der jeweiligen Module dokumentiert. Die Abfragen sind in der Datenbank in der Tabelle maskeninfo sowie felderinfo; Änderungen sind im Howto dokumentiert. Die Erzeugung von Hilfedokumenten für die Abfragen ist im Abschnitt Javahelp beschrieben.

Das Applet und Servlet wurde in Java programmiert. Änderungen werden im Folgenden beschrieben.

### 5.1Kompilieren der Java-Quellen

Das Java-Applet und das Servlet sind im Quellcode verfügbar. Für die Entwicklung nutzen wir das Build-Tool Ant, es können aber auch andere Entwicklungsumgebungen eingesetzt werden.

Die SuperX-Quellen haben folgende Struktur:

| | |
|---|---|
| **de.superx.applet** | Klassen des SuperX-Applets |
| **de.superx.dbadmin** | Klassen des SuperX-Admintools |
| **de.superx.servlet** | Klassen des SuperX-Servlet |
| **de.superx.bin** | Kommandozeilen-Klassen für den SuperX-Client |
| **de.superx.util** | Gemeinsam benutzte Dateien |
| **de.memtext.*** | Gemeinsam benutzte Utilities der Fa. memtext |
| **images** | Gemeinsam benutzte Grafiken |
| **com.sun.help** | Javahelp-Klassen |
| **javax.help.** | Ebenfalls Javahelp-Klassen |

Auf dem Webserver wird im Verzeichnis
`$SUPERX_DIR/webserver/tomcat/webapps/superx/WEB-INF/lib`
die Datei superx<<Versionsnr.>>.jar abgelegt. Diese enthält alle Klassen. Beim Zugriff über Tomcat ist die Datei automatisch im CLASSPATH, beim SuperX-Client via `jdbc` muss diese Datei manuell, z.B. über die Datei SQL_ENV, im CLASSPATH sein (Umgebungsvariable JDBC_CLASSPATH).

Das Applet und das Admintool soll wegen WWW-Einsatz möglichst "leicht" sein, deshalb wird es separat kompiliert. Lediglich die Klasse `de.superx.servlet.SuperX_el.class` wird sowohl vom Applet als auch vom AdminTool benötigt. Das Archiv heißt jeweils `superx.jar` für das Applet, und `SuperXDBAdmin.jar` für das Admintool. Beide werden nach
`$SUPERX_DIR/webserver/tomcat/webapps/superx/applet`
kopiert.

Beachten Sie beim Kompilieren, dass das Applet und das AdminTool bei vielen Browsern im Cache gehalten wird (selbst wenn der Browser immer nach aktuellen Versionen suchen soll). Sie sollten nach neuem Kompilieren sicherheitshalber immer den Cache löschen und den Browser einmal beenden. Alternativ können Sie das Applet auch lokal aus dem Browser starten (also nicht über http://), Sie müssen lediglich eine korrekt eingestellte `superx.properties` mit gültiger `SxServerURL` im gleichen Verzeichnis haben.

### 5.1.1Kompilieren mit Bordmitteln des JDK

Aufgrund der Komplexität der eingebundenen Klassen ist ein Build mit normalen Bordmitteln des JDK zwar möglich, aber viel zu umständlich. Der Build läuft voreingestellt nur mit ANT (s.u.) und unter Linux.

### 5.1.2Kompilieren mit dem Jakarta-Build-Tool ant

Wir empfehlen, Applet und Servlet mit dem im Kernmodul enthaltenen Werkzeug `ant` zu kompilieren, das bereits in dem SuperX-Kernmodul enthalten ist. Sämtliche Quellen lassen sich von der Konsole aus mit dem Sun JDK 1.4.x und ANT kompilieren. Folgende Pfade sind für Entwickler wichtig:

| | |
|---|---|
| **Javadoc-Dateien zum gesamten SuperX-Paket** | `$SUPERX_DIR/doc/apidoc` |
| **Quellcode des SuperX-Applets** | `$SUPERX_DIR/webserver/tomcat/webapps/superx/WEB-INF/src/de/superx/applet` |
| **Quellcode des SuperX-Servlets** | `$SUPERX_DIR/webserver/tomcat/webapps/superx/WEB-INF/src/de/superx/servlet` |
| **Quellcode des SuperX-Admintools** | `$SUPERX_DIR/webserver/tomcat/webapps/superx/WEB-INF/src/de/superx/dbadmin` |
| **Ant-Pfad zur build-xml** | `$SUPERX_DIR/webserver/tomcat/webapps/superx/WEB-INF/src/build.xml` |
| **Ant-Shellscript für den Build** | `$SUPERX_DIR/webserver/tomcat/webapps/superx/WEB-INF/src/build_it.x` |

Die Quellen lassen sichmit einem JDK-Compiler der Generation 1.4.x oder höher kompilieren. Zum Kompilieren des Servlets muss die Bibliothek für Servlets im Classpath enthalten sein; dies ist bei der normalen SuperX-Distribution der Fall.

Bei einer eingerichteten Umgebung für SuperX brauchen Sie die `build.xml` nicht anpassen. Bei benutzerspezifischen Einstellungen passen Sie die Einträge zum CLASSPATH, zur SuperX-Version und zu SUPERX_DIR an. Für die Versionierung wird der Filter-Mechanismus in ant genutzt, d.h. jedesVorkommen des Strings "@version@" wird durch den aktuellen Wert ersetzt, der in der ant-Property VERSION gesetzt ist.

Zur Nutzung von `ant` wechseln Sie in der Konsole in das Verzeichnis
`$SUPERX_DIR/webserver/tomcat/webapps/superx/WEB-INF/src`
und rufen Ant wie folgt auf:

| | |
|---|---|
| **Kompilieren der Klassen des Applets** | `build_it.x compileApplet` |
| **Kompilieren und Erzeugen des Applets** | `build_it.x distApplet` |
| **Dokumentieren des Applets** | `build_it.x docApplet` |
| **Kompilieren des gesamten Pakets** | `build_it.x compileServer` |
| **Erzeugen der superx<<Versionsnr>>.jar** | `build_it.x distServer` |
| **Dokumentieren des Servlets** | `build_it.x docServlet` |
| **Kompilieren der Klassen des Admin-Tools** | `build_it.x compileAdmin` |
| **Kompilieren und Erzeugen des Admin-Tools** | `build_it.x distAdmin` |
| **Dokumentieren des Admin-Tools** | `build_it.x docAdmin` |

Bei der Distribution des Applets ist im ant-Script folgende Nachbearbeitung vorgesehen: zunächst werden nicht benötigte Klassen aus der superx.jar entfernt, um das Applet möglichst klein zu halten. Das OpenSource-Tool `obfuscator` wird aufgerufen und die resultierende `superx.jar` wird an die richtige Stelle kopiert ($SUPERX_DIR/webserver/tomcat/webapps/superx/applet). Danach ist eine Signierung des Applets vorgesehen. Die Syntax ist in dem Kommentar des targets `distApplet` in der `build.xml` beschrieben:

Geben Sie auf der Kommandozeile ein:

| | |
|---|---|
| **Befehlsfolge zum Signieren des Applets** | `keytool -genkey -alias superx_applet -keyalg RSA`<br>`keytool -selfcert -alias superx_applet -validity 365` |

Als Passwort wählen Sie das, das in der build.xml vorgesehen ist. Der Wert hinter Validitiy beschreibt den Gültigkeitszeitraum des Zertifikats (in Tagen).

Wenn Sie das Zertifikat erneuern wollen, müssen Sie es zunächst löschen mit

```
keytool -delete -alias superx_applet
```

Das gleiche Vorgehen gilt für das Admin-Tool.

### 5.1.3Entwicklung mit Jedit

Als Entwicklungsumgebung empfehlen wir Eclipse von IBM oder den plattformübergreifend verfügbaren OpenSource-Editor jedit (www.jedit.org). Er unterstützt via Pugins die Java-Entwicklung. Für SuperX benötigen Sie die folgenden Plugins:

| **Plugins für Jedit** | Console-Plugin<br>JBrowse<br>JCompiler<br>AntFarm<br>XML<br>XSLT |
|---|---|

Im Clientpaket sind diese Plugins bereits enthalten.

Sie starten das Plugin `AntFarm`, und geben als Build-File die Datei `$SUPERX_DIR/webserver/tomcat/webapps/superx/WEB-INF/src/build.xml` an. Die Targets werden dann übersichtlich angezeigt und können sofort ausgeführt werden.

## 5.2Erzeugung der SuperX-Hilfe im Javahelp-Format

Die SuperX-Hilfe besteht aus einem Archiv im Javahelp-Format. Sie ist nur für das Applet nutzbar. Die Hilfetexte sind in den Modulen erzeugt und können problemlos integriert werden. Falls Sie eigene Hilfetexte einbinden wollen, müssen Sie wie folgt vorgehen:

1. Erzeugen Sie html-Seiten mit der Hilfe (html 3.2)
2. Binden Sie die Dateien in die Mapping-Datei ein (`$SUPERX_DIR/webserver/tomcat/webapps/superx/applet/javahelp/map.jhm`)
3. Falls die Hilfeseiten kontextabhängig abrufbar sein sollen, müssen die Titel der Mapping-Einträge folgenden Konventionen folgen:

- Allgemeine Beschreibungen der Abfragen lauten A<<TID>>.htm
- Beschreibungen der Masken lauten M<<TID>>.htm
- Beschreibungen der Ergebnistabellen lauten T<<TID>>.htm

Am Anfang ist es hilfreich, die vorhandenen Hilfetexte als Vorlage zu benutzen.

Die Javahilfe kann auch komfortabler mit dem Memtext-Autorensystem aus einer Word-Datei erzeugt werden. Details dazu siehe http://studio.memtext.de.

## 5.3 Versionshistorie

4.0rc1 01-2010

**Entwickler/innen** | Meikel Bisping, Daniel Quathamer, Andre Knieschewski

- Redesign der XML-Oberfläche: Menüs mit Erläuterungstexten, Breadcrumbs
- Postgres 8.3 oder 8.4 Unterstützung
- UTF-8 Release

3.5rc1 6/2008

**Entwickler/innen** | Meikel Bisping, Daniel Quathamer

- Performanceverbesserung beim Maskenaufbau (Feldart1-Cache)
- Layoutanpassungen

3.5rc1 (3/2008)

- Neue Maske **Gruppe kopieren**
- **Ajax-Client** für das XML-Frontend
- Verbessertes **Benutzer-Handbuch** für das XML-Frontend
- Verbesserte **Administrations-Masken**
- Verbesserter **PDF-Export**: automatische Spaltenbreiten-Skalierung (1 Seite Querformat), mehrdimensionale Ergebnisspalten
- Verbesserter **Excel-Export**: mehrdimensionale Ergebnisspalten
- **RTF-Export** wurde gestrichen, weil OSS-Bibliothek JFor veraltet ist

3.5beta (10/2007)

**Entwickler/innen** | Meikel Bisping, Daniel Quathamer, Christoph Litz

- Neuprogrammierung des XML-Frontends ("Ajax-Client")
- Nach Login kann Hinweis-Seite angezeigt werden (z.B. für Datenschutz-Hinweise

3.0 final (05/2007)

**Entwickler/innen** | Meikel Bisping, Daniel Quathamer, Christoph Litz

- Viele neue Sicherheitsfeatures (Passwort-Policy etc.)
- Verbessertes XML-Frontend: Excel-Export, Baummenüs etc.
- Verbesserte Administrationsmasken (DBFORMS) zur Userverwaltung etc.

3.0 beta (04/2005)

**Entwickler/innen** | Meikel Bisping, Daniel Quathamer

- Neue Stored Procedures für Postgres- Organigramm-Auswertung
- Abbildung alternativer Hierarchien und Anbindung an Userverwaltung
- Mandantenfähigkeit
- Einsatz von Freemarker als Template Engine für Masken-Scripte (und damit Java-Unterstützung der Scripte), erste Libraries für Postgres und Informix-unabhängigen Code
- Glossare und Felderläuterungen abrufbar
- Einsatz von dbforms 2.5 als Formular-Engine; erste Administrationsformulare
- Komplettes Refactoring des XML-Frontend inkl. Cacheing, Organigramm-Darstellung, XSL-Mechanismen

2.1 (04/2004)

**Entwickler/innen** | Meikel Bisping, Daniel Quathamer, Marlies Winterstein

- SSL/Apache-Anbindung dokumentiert, Musterdateien für `mod_jk` fertig
- XML-Frontend liefert html,- XML und text-Export sowie rtf und pdf; anderer Authentifizierungsmechanismus (Cookie)
- Stylesheet-Verwaltung verbessert, erste Beispielstylesheets für `lynx`.
- Unter Postgres im XML-Frontend sind Masken zur Administration von Masken und Usern fertiggestellt
- Bugfixes im XML-Frontend (Pflichtfelder abfangen, Feldinhalte einlesen)
- Admin-Tool 0.93 mit vielen Bugfixes (insbes. für Postgres)
- Applet: Implementation von Organigramm-"Sichten" (noch nicht dokumentiert);
  viele kleine Verbesserungen, z.B. beim Anmeldedialog, Tabellendarstellung, Bedienung.
- Tomcat4-Anbindung
- Datenbankschema in XML auf der Basis von Apache Torque-DTD und HIS-DTD (ergänzt um eigene Tags für ETL-Prozesse). Die Datenbank selbst wurde um die Tabellen `db_version` und `db_tabellen` ergänzt, um HIS-konform zu arbeiten. Aus den XML-Dateien werden die Modulscripte und Dokumentationen erzeugt.
- Neue Stored Procedures für Anmeldedialog (`sp_user_themen`). Makros werden im Applet jetzt ausgeblendet.
- Neues Installationsscript für Datenbank; diverse Anleitungen für RedHat,- SuSE Linux und Cygwin.
- Installationsscripte für Modulinstallation, Aktualisierung, Deinstallation (alpha); neue Shellscripte zum ETL-Prozeß
- jdbc-Client für Kommandozeile fertiggestellt (DOS und UNIX)
- Neue Kommandozeilen-Scripte unter DOS und Unix:
- Maskenverwaltung
- Tabellenextraktion / Upload
- Datenbankschemata von Tabellen
- XML-Transformation
- Konvertierung von Rohdaten-Dateien nach der Maßgabe von Import/Exportspezifikationen
- Postgres 7.3 oder 7.4 wird unterstützt
- Verbessertes Access-Frontend: Formulare für alle relevanten Tabellen des Kernmoduls

2.01 (06/2003)

**Entwickler** | Meikel Bisping, , Marlies Winterstein, Daniel Quathamer

- Integration der Javahilfe ins Applet
- Signierung des Applets -> Keine Client-Installation außer JRE mehr notwendig
- Aufbau der Package de.superx.*
- Update auf JRE 1.4 in html-Aufrufseiten
- Java-Installationswebsite für versch. Browser verbessert (JSP-Seite mit Anpassung für Netscape 6.x / 7.x, Mozilla 1.3.x, IE 5.x,6.x)

2.0 (03/2002)

**Entwickler** | Marlies Winterstein, Meikel Bisping, Daniel Quathamer

- Einbettung von kontextabhängigen Hilfeseiten mit Javahelp
- Stabilität und Performance im Netzbetrieb durch Connection Pooling
- Entwurf eines Werkzeugs zur Administration von Organigramm und Userrechten ("SuperX-Admintool")

- Fertigstellung eines Prototypen zur Administration via MS Access 2000
- Beliebig "tief" verschachtelbares und zeitabhängiges Organigramm
- Modularisierung von SuperX
- Baumstruktur im Organigramm eingebaut
- Baumstruktur im Themenbaum
- Stored Procedures für Abbildung der hierarchisches Struktur des Organigramms
- Modularisierung von SuperX vollzogen
- Applet-Servlet-Struktur
- XML-Frontend mit Makrofunktion
- -Möglichkeiten sicherer Verbindungen Servlet-Applet
- Fertigstellung eines Prototypen des Kernmoduls auf der Basis von PostgreSQL 7.2

1.0 (04/2001)

| **Entwickler** | Reiner Behr (Uni Karlsruhe) |
|---|---|

- Portierung des Win32-Client nach Java
- Datenbankzugriff über jdbc
- Ergebnistabellen optimiert (Sortierung / Löschung von Spalten, Druckfunktion)